Московский физико-технический институт
(Государственный Университет)

На правах рукописи

УДК 519.22

Ивахненко Андрей Александрович

Комбинаторные оценки вероятности переобучения и их применение в логических алгоритмах классификации

Специальность 01.01.09 — дискретная математика и математическая кибернетика

Диссертация на соискание учёной степени
кандидата физико-математических наук

Научный руководитель: д. ф.-м. н. Воронцов Константин Вячеславович

Москва
2010

Оглавление

Введение

Диссертационная работа относится к математической теории распознавания и классификации и посвящена проблеме повышения обобщающей способности логических алгоритмов классификации, основанных на поиске информативных конъюнктивных закономерностей в массивах прецедентных данных с вещественными, порядковыми и номинальными признаками.

Актуальность темы. Логические алгоритмы классификации широко используются для автоматизации принятия решений в трудноформализуемых областях, таких как медицинская диагностика, геологическое прогнозирование, кредитный скоринг, направленный маркетинг и т. д. Их преимуществом является возможность содержательной интерпретации как внутреннего строения алгоритма, так и принимаемых им решений на естественном языке в терминах предметной области. Однако качество классификации (обобщающая способность) логических алгоритмов, как правило, немного уступает более сложным конструкциям, таким как бустинг или бэггинг над решающими деревьями, которые являются «чёрными ящиками» и не обладают свойством интерпретируемости. Поэтому актуальной задачей является повышение обобщающей способности логических алгоритмов классификации без существенного усложнения их внутреннего строения.

Стандартные подходы теории статистического обучения дают слишком осторожные, пессимистичные оценки обобщающей способности. Эффекты переобучения, возникающие при поиске логических закономерностей, являются относительно слабыми. Они существенно зависят от конкретной выборки данных и потому плохо описываются стандартными оценками. Недавно разработанная комбинаторная теория переобучения даёт существенно более точные оценки вероятности переобучения. Актуальность данной диссертационной работы связана ещё и с тем, что она является первым примером практического применения комбинаторной

теории переобучения.

Цель работы — получение комбинаторных оценок обобщающей способности логических закономерностей и разработка на их основе новых методов повышения качества логических алгоритмов классификации.

Научная новизна. Получена новая комбинаторная оценка вероятности переобучения, зависящая от характеристик расслоения и связности семейства алгоритмов. Комбинаторные оценки, полученные ранее другими авторами, либо были существенно менее точными, либо опирались на сильные дополнительные предположения о существовании в семействе корректного или хотя бы единственного лучшего алгоритма, что почти невозможно гарантировать на практике.

Впервые описана структура классов эквивалентности конъюнктивных логических закономерностей при разнотипных исходных данных, включающих количественные, порядковые и номинальные признаки.

Предложен новый метод вычисления поправки на переобучение, позволяющий улучшить широкий класс стандартных критериев информативности логических закономерностей.

Методы исследования. Для получения оценок вероятности переобучения использована слабая (перестановочная) вероятностная аксиоматика, комбинаторная теория переобучения, элементы комбинаторки, теории вероятностей и теории графов. Для проверки точности комбинаторных оценок проведены вычислительные эксперименты на модельных данных. Для практического применения полученных оценок вероятности переобучения предлагается встраивать вычисление поправок на переобучение в стандартные критерии информативности, что позволяет усовершенствовать широкий класс эвристических методов построения логических алгоритмов классификации. Для проверки данного подхода проведены эксперименты на реальных задачах классификации.

Положения, выносимые на защиту.

1. Общая комбинаторная оценка вероятности переобучения, зависящая от характеристик расслоения и связности семейства алгоритмов.
2. Описание структуры классов эквивалентности семейства пороговых конъюнкций над количественными, порядковыми и номинальными признаками.
3. Эффективный метод вычисления поправок на переобучение в критериях информативности для широкого класса эвристических алгоритмов поиска логических закономерностей.
4. Экспериментальное подтверждение того, что предложенный метод приводит к повышению обобщающей способности алгоритмов классификации, представляющих собой композиции логических закономерностей.

Теоретическая значимость. Данная работа вносит существенный вклад в развитие комбинаторной теории переобучения и является первым успешным примером практического применения комбинаторных оценок вероятности переобучения.

Практическая значимость. Предложенные методы расширяют область применимости логических алгоритмов классификации и повышают качество решения задач классификации в тех предметных областях, где применение логических алгоритмов продиктовано соображениями интерпретируемости.

Апробация работы. Результаты работы докладывались, обсуждались и получили одобрение специалистов на следующих научных конференциях и семинарах:

- Всероссийская конференция «Математические методы распознавания образов» ММРО-12, 2005 г. [24];
- Международная конференция «Интеллектуализация обработки информации» ИОИ-6, 2006 г. [11];

- 49-я научная конференция МФТИ, 2006 г. [18];
- Всероссийская конференция «Математические методы распознавания образов»ММРО-13, 2007 г. [22, 25, 26];
- Международная конференция «Интеллектуализация обработки информации» ИОИ-7, 2008 г. [12];
- Всероссийская конференция «Математические методы распознавания образов» ММРО-14, 2009 г. [13].
- 52-я научная конференция МФТИ, 2009 г. [19];
- Международная конференция «Интеллектуализация обработки информации», ИОИ-8, 2010 г. [20, 28].
- Международная конференция «Распознавание образов и анализ изображений: новые информационные технологии», РОАИ-10, 2010 г. [58].
- Научные семинары отдела Интеллектуальных систем Вычислительного центра РАН и кафедры «Интеллектуальные системы» МФТИ, 2006 – 2010 г.г.

Публикации по теме диссертации. Всего публикаций по теме диссертации — 13, в том числе в изданиях из Списка, рекомендованного ВАК РФ — одна [21].

Краткое содержание работы по главам. Работа состоит из четырёх глав.

В первой главе вводятся определения и обозначения, необходимые для дальнейшего рассмотрения задач классификации, проблемы переобучения и логических алгоритмов классификации. Приводится краткий обзор литературы по данным вопросам. Глава завершается формальной постановкой основной задачи данной работы — получения оценок вероятности переобучения для логических закономерностей и использования этих оценок для улучшения обобщающей способности логических алгоритмов классификации.

Во второй главе предлагается новая оценка вероятности переобучения, учитывающая свойства расслоения и связности семейства алгоритмов. Эта оценка, исходно сформулированная для алгоритмов классификации, распространяется на параметрическое семейство конъюнктивных правил. Описывается структура классов эквивалентности данного семейства и эффективный алгоритм вычисления оценки расслоения–связности. Предлагается использовать эту оценку для модификации стандартных критериев информативности правил. Модификация сводится к введению «поправки на переобученность». Приводятся результаты экспериментов на модельных данных, показывающие, что модифицированные критерии, в отличие от стандартных, позволяют избежать переобучения, возникающего в результате оптимизации порогов в термах, составляющих конъюнкцию.

В третьей главе описываются алгоритмы, реализованные автором в рамках библиотеки логических алгоритмов классификации Forecsys Logic Pro, в их числе: градация вещественных признаков, синтез информативных закономерностей, построение множеств Парето-оптимальных закономерностей и расслоения Парето, построение композиций закономерностей.

В четвёртой главе приводятся результаты экспериментов на реальных данных, показывающие, что применение критериев информативности с поправкой на переобучение, как правило, улучшает обобщающую способность логических алгоритмов классификации.

Благодарности

Автор выражает глубокую признательность научному руководителю д. ф.-м. н. Константину Вячеславовичу Воронцову за постановку задачи и постоянную поддержку в ходе исследований, заведующему кафедрой «Интеллектуальные системы» чл.-корр. РАН Константину Владимировичу Рудакову за ценные замечания, к. ф.-м. н. Вадиму Викторовичу Стрижову за внимание к работе, помощь в поиске литературы и подборе данных для экспериментов, к. ф.-м. н. Ольге Сергеевне Федько за организационную помощь, коллегам по работе и учёбе в ас-

пирантуре МФТИ за плодотворные обсуждения работы и моральную поддержку.

Глава 1. Проблема обобщающей способности в логических алгоритмах классификации

В данной главе вводятся основные понятия и обозначения, относящиеся к задачам классификации, логическим алгоритмам классификации, проблеме переобучения, и формулируется основная цель исследования.

1.1 Задача классификации

Пусть имеется пространство *объектов* $\mathbb{X}$ и конечное множество *классов* $\mathbb{Y}$. Объекты описываются набором n *признаков* $f_j\colon \mathbb{X} \to D_j$, $j = 1, \ldots, n$, где D_j — множество допустимых значений j-го признака. Таким образом, произвольному объекту $x \in \mathbb{X}$ соответствует *признаковое описание* $\big(f_1(x), \ldots, f_n(x)\big)$. В зависимости от множества D_j признаки делятся на типы:

- бинарные ($D_j = \{0, 1\}$),
- номинальные (D_j — конечное множество),
- порядковые (D_j — конечное упорядоченное множество),
- количественные ($D_j = \mathbb{R}$).

Целевой признак $y^*\colon \mathbb{X} \to \mathbb{Y}$ известен только на объектах *обучающей выборки* $X = \{x_i\}_{i=1}^{\ell} \subset \mathbb{X}$, $y_i = y^*(x_i)$. Требуется построить *алгоритм классификации* — функцию $a\colon \mathbb{X} \to \mathbb{Y}$, которая по признаковому описанию произвольного объекта x предсказывала бы его класс $y^*(x)$, то есть приближала бы неизвестную функцию y^* на всём множестве $\mathbb{X}$.

Задана бинарная функция $I(a, x)$, называемая *индикатором ошибки*. Если $I(a, x) = 1$, то говорят, что алгоритм a ошибается на объекте x. Обычно полагается $I(a, x) = [a(x) \neq y^*(x)]$.

Определим *число ошибок* алгоритма a на выборке $X \subset \mathbb{X}$

$$n(a, X) = \sum_{x \in X} I(a, x)$$

и *частоту ошибок* алгоритма a на выборке X

$$\nu(a, X) = \frac{1}{|X|} n(a, X).$$

Методом обучения называется отображение вида $\mu\colon 2^{\mathbb{X}} \to A$, которое произвольной обучающей выборке $X \subseteq \mathbb{X}$ ставит в соответствие некоторый алгоритм $a = \mu X$ из A. Метод μ называется методом *минимизации эмпирического риска* (МЭР), если

$$\mu X \in A(X) = \operatorname{Arg\,min}_{a \in A} n(a, X);$$

и *пессимистичным* методом МЭР, если

$$\mu X = \arg \max_{a \in A(X)} n(a, \mathbb{X});$$

Пессимистичный метод МЭР не реализуем на практике, т. к. контрольная выборка скрыта в момент обучения. Тем не менее, он представляет значительный теоретический интерес, так как точные оценки вероятности переобучения для пессимистичного метода МЭР являются достижимыми верхними оценками для произвольного метода МЭР.

1.2 Проблема переобучения

Если алгоритм a доставляет минимум функционалу $\nu(a, X)$ на заданной обучающей выборке X, то это ещё не гарантирует, что он будет хорошо обобщать эмпирические данные X, то есть приближать целевую зависимость на произвольной контрольной выборке $\bar{X} = (x'_i, y'_i)_{i=1}^k$. Когда качество работы алгоритма

на новых объектах, не вошедших в состав обучения, оказывается существенно хуже, чем на обучающей выборке, говорят, что имеет место *переобучение* или переподгонка (overfitting).

Разобъём выборку $\mathbb{X}$ всеми C_L^ℓ способами на две непересекающиеся подвыборки: наблюдаемую обучающую X длины ℓ и скрытую контрольную $\bar{X}$ длины $k = L - \ell$, неизвестную в момент обучения. Следуя слабой вероятностной аксиоматике [54], сделаем единственное вероятностное предположение, что все C_L^ℓ разбиений равновероятны. Данное предположение фактически эквивалентно стандартному предположению о независимости наблюдений в двух подвыборках. Определим для любого $\varepsilon > 0$ *вероятность переобучения* метода μ как

$$Q_\varepsilon(\mu, \mathbb{X}) = \mathsf{P}\big[\nu(\mu X, \bar{X}) - \nu(\mu X, X) \geqslant \varepsilon\big], \tag{1.1}$$

где знак вероятности можно понимать как среднее по всем разбиениям: $\mathsf{P} \equiv \frac{1}{C_L^\ell} \sum_{X \sqcup \bar{X}}$.

Коль скоро такая оценка получена, можно утверждать, что с вероятностью $1 - \eta$, достаточно близкой к единице,

$$\nu(\mu X, \bar{X}) < \nu(\mu X, X) + \varepsilon(\eta), \tag{1.2}$$

где $\varepsilon(\eta)$ — функция, обратная к $\eta(\varepsilon) = Q_\varepsilon(\mu, \mathbb{X})$.

Рассмотрим сначала простейший частный случай, когда семейство состоит из единственного алгоритма, $A = \{a\}$, допускающего известное число $m = n(a, \mathbb{X})$ ошибок на полной выборке. Разумеется, метод обучения для любой выборки X возвращает этот алгоритм, $\mu X \equiv a$. В таком случае справедлива точная оценка функционала Q_ε через функцию гипергеометрического распределения:

$$Q_\varepsilon(a, \mathbb{X}) = \mathsf{P}\big[\nu(a, \bar{X}) - \nu(a, X) \geqslant \varepsilon\big] = \mathsf{P}\big[n(a, X) \leqslant s_m(\varepsilon)\big] = H_L^{\ell,m}\left(s_m(\varepsilon)\right),$$

где $s_m(\varepsilon) = \frac{\ell}{L}(m - \varepsilon k)$ — максимальное число ошибок на обучающей выборке, при котором алгоритм a «переобучен», $H_L^{\ell,m}\left(s_m(\varepsilon)\right)$ — функция гипергеометрического

распределения:

$$h_L^{\ell,m}(s) = \mathsf{P}\big[n(a,X)=s\big] = \frac{C_m^s C_{L-m}^{\ell-s}}{C_L^\ell};$$
$$H_L^{\ell,m}(z) = \mathsf{P}\big[n(a,X)\leqslant z\big] = \sum_{s=0}^{\lfloor z\rfloor} h_L^{\ell,m}(s)\,.$$

Для общего случая, когда множество A произвольно, в теории статистического обучения известны различные оценки обобщающей способности, см. обзоры [7, 33]. Первые верхние оценки, полученные В. Н. Вапником и А. Я. Червоненкисом [4, 5, 6] основывались на принципе равномерной сходимости частот в двух подвыборках:

$$Q_\varepsilon(\mu,\mathbb{X}) \leqslant \bar{Q}_\varepsilon(\mu,\mathbb{X}) = \mathsf{P}\big[\max_{a\in A}\big(\nu(a,\bar{X}) - \nu(a,X)\big) \geqslant \varepsilon\big].$$

Замена функционала Q_ε его верхней оценкой $\bar{Q}_\varepsilon$ может приводить к заметной потере точности оценки. С другой стороны, благодаря этому приёму получаются оценки вида (1.2), справедливые для любого метода обучения μ. Это позволяет минимизировать правую часть (1.2) и тем самым получить новый метод обучения, отличный от минимизации эмпирического риска, и называемый *структурной минимизацией риска*. Добавка $\varepsilon(\eta)$ в теории Вапника-Червоненкиса представляет собой штраф за сложность семейства алгоритмов A. Чем выше сложность, тем больше значение штрафа $\varepsilon(\eta)$. Чтобы воспользоваться методом структурной минимизации риска, необходимо заранее — до того, как станет известна обучающая выборка X — фиксировать систему вложенных подсемейств $A_1 \subseteq A_2 \subseteq \cdots \subseteq A$, и затем выбрать то подсемейство A_h, при котором правая часть (1.2) минимальна для заданной обучающей выборки X. Таким образом, структурная минимизация риска — это поиск подсемейства минимальной сложности, содержащего алгоритм с приемлемо низким числом ошибок на обучающей выборке.

Сложностные оценки Вапника-Червоненкиса завышены на несколько порядков, как показывают численные эксперименты [54, 56]. Это происходит из-за того, что в ходе вывода оценки применяется неравенство Буля — вероятность

объединения событий оценивается сверху суммой их вероятностей:

$$\begin{aligned}\bar{Q}_\varepsilon(\mu, \mathbb{X}) = \mathsf{P} \max_{a \in A} \big[\nu(a, \bar{X}) - \nu(a, X) \geqslant \varepsilon\big] \leqslant \\ \leqslant \sum_{a \in A} \mathsf{P}\big[\nu(a, \bar{X}) - \nu(a, X) \geqslant \varepsilon\big] \leqslant \\ \leqslant \sum_{a \in A} H_L^{\ell,m}\left(s_m(\varepsilon)\right),\end{aligned}$$

где $m = n(a, \mathbb{X})$, $s_m(\varepsilon) = \frac{\ell}{L}(m - \varepsilon k)$. Таким образом, оценка Вапника-Червоненкиса представляет собой сумму оценок $Q_\varepsilon(a, \mathbb{X})$ по всем алгоритмам $a \in A$.

В дальнейшем оценки Вапника-Червоненкиса неоднократно уточнялись [53] и создавались принципиально иные подходы, не использующие неравенство Буля, в частности, теория рademахеровской сложности [32] и PAC-байесовский подход [47]. Оценки, получаемые в рамках данных теорий, также завышены, хотя и в меньшей степени. Новые подходы существенно используют континуальные вероятностные меры и отходят от исходной комбинаторной техники получения оценок, которую применяли Вапник и Червоненкис. Постепенное усложнение теории переобучения приводит к тому, что разобраться в причинах завышенности и эффективно устранить их становится всё труднее.

Данная работа основана на комбинаторной теории переобучения [10, 57], которая также не использует неравенство Буля, но позволяет экспериментально измерять факторы завышенности оценок, учитывать эффекты расслоения и связности в семействах алгоритмов, получать оценки, достаточно близкие к точным, а в отдельных случаях — точные оценки.

1.3 Логические алгоритмы классификации

Логические алгоритмы классификации представляют собой композиции информативных логических правил.

Логическое правило — это функция вида $r\colon \mathbb{X} \to \{0, 1\}$ из некоторого параметрического семейства функций R. Обычно к правилам предъявляют требование

интерпретируемости: правило должно зависеть от небольшого числа признаков и допускать запись на естественном языке. В данной работе рассматриваются правила, имеющие вид конъюнкций пороговых предикатов:

$$r(x) \equiv r(x; c^1, \ldots, c^n) = \prod_{j \in \omega} \left[x^j \leqslant_j c^j \right], \tag{1.3}$$

где $x = (x^1, \ldots, x^n) \in \mathbb{R}^n$ — признаковое описание объекта x, $\omega \subseteq \{1, \ldots, n\}$ — подмножество признаков, $c^j \in D_j$ — порог по j-му признаку, $\leqslant_j$ — одна из операций сравнения: $\{\leqslant, \geqslant\}$ для количественных и порядковых признаков, $=$ для номинальных. Выражение $[x^j \leqslant_j c^j]$ будем называть *термом* j-го признака.

Данное семейство правил широко применяется при решении практических задач классификации [27, 17, 3, 16, 31, 24, 15, 25, 26, 9].

Говорят, что правило r *выделяет* объект x, если $r(x) = 1$.

Логическая закономерность — это правило, удовлетворяющее требованию *информативности* — оно должно выделять достаточно много объектов одного из классов $y \in \mathbb{Y}$, и практически не выделять объекты других классов. Поиск закономерностей класса y по обучающей выборке $X \subset \mathbb{X}$ в семействе $r \in R$ сводится к решению задачи двухкритериальной оптимизации:

$$P_y(r, X) = \frac{1}{|X|} \sum_{x_i \in X} r(x_i) \, [y_i = y] \to \max;$$
$$N_y(r, X) = \frac{1}{|X|} \sum_{x_i \in X} r(x_i) \, [y_i \neq y] \to \min;$$

На практике двухкритериальную задачу заменяют однокритериальной и оптимизируют некоторый критерий информативности $H(P, N)$. Известны десятки различных критериев: энтропийный, индекс Джини, точный тест Фишера, тест χ^2, тест ω^2 и другие [48]. Большинство критериев оценивают степень неслучайности разбиения обучающей выборки X на два подмножества (положительные примеры x: $r(x) = 1$ и отрицательные x: $r(x) = 0$) относительно исходного разбиения выборки X на классы. Однако ни один из критериев нельзя назвать безусловно предпочтительным. Выбор критерия информативности является эвристикой.

Поскольку каждая закономерность выделяет лишь часть выборки, алгоритм классификации строится как композиция закономерностей. Одной из наиболее распространённых конструкций является *взвешенное голосование* закономерностей:

$$a(x) = \arg\max_{y \in Y} \sum_{r \in R_y} w_r r(x),$$

где R_y — множество правил класса y, $w_r \geqslant 0$ — вес правила r. Построение таких композиций можно вести по-разному.

Одна из стратегий заключается в том, чтобы сначала построить списки закономерностей R_y, затем, рассматривая правила $r(x)$ как новые признаки, построить на них линейный классификатор любым из известных методов, например, логистической регрессией или методом опорных векторов [44].

Другая стратегия заключается в том, чтобы строить правила по очереди, и для каждого правила r сразу определять вес w_r. Так, в частности, работает алгоритм бустинга [36].

1.4 Постановка задачи

Чтобы алгоритм классификации имел высокую обобщающую способность, составляющие его закономерности не должны быть переобучены.

Недостаток стандартных критериев информативности в том, что они не учитывают переобучение. При оптимизации $P(r, X)$ и $N(r, X)$ по обучающей выборке X соответствующие величины $P' = P(r, \bar{X})$ и $N' = N(r, \bar{X})$ уже не будут оптимальны на контрольной выборке $\bar{X} = \mathbb{X} \backslash X$.

В литературе известны две методики учёта переобучения в логических алгоритмах классификации. Первая — эмпирическая методика мета-обучения, описанная в серии работ Фюрнкранца и др. [41, 42, 45, 46]. Идея заключается в том, чтобы для большого числа (десятков) реальных задач классификации найти большое число (порядка сотен тысяч) правил и восстановить регрессионную зависимость величин P', N' на контроле от P, N на обучении. При этом в роли обучающих

объектов выступают все найденные правила. Для восстановления зависимости используется либо полиномиальная регрессия, либо нейронная сеть. Недостатком данного подхода является то, что ищется некоторая универсальная зависимость, не учитывающая особенности конкретной выборки данных.

Вторая методика основана на применении теоретических верхних оценок вероятности переобучения. Оценки [14, 54], основанные на теории Вапника-Червоненкиса [6], позволяют связать вероятность переобучения со сложностью семейства правил R. Они зависят от ранга конъюнкции $|\omega|$ и числа допустимых значений признаков x^j по каждой размерности $j \in \omega$, но не зависят от конкретной выборки данных $\mathbb{X}$. Согласно экспериментам [54] эти оценки сильно завышены, что делает их непригодными для количественного предсказания значений (P', N').

Точные комбинаторные оценки вероятности переобучения [56, 57] позволяют учитывать свойства расслоения и связности — более тонкие характеристики семейства, зависящие от данных $\mathbb{X}$. До сих пор точные оценки удавалось получать лишь для некоторых модельных семейств алгоритмов: монотонных и унимодальных цепочек [57], монотонных и унимодальных многомерных сеток [1, 2], связок монотонных цепочек [39], интервалов и шаров булева куба [29], и некоторых других. Для реальных семейств алгоритмов классификации точных комбинаторных оценок вероятности переобучения пока не известно.

Первая задача, которая ставится в данной работе — получить комбинаторные оценки вероятности переобучения для семейства правил вида (1.3).

Вторая задача — применить полученные оценки для улучшения стандартных критериев информативности в логических алгоритмах классификации.

Глава 2. Комбинаторные оценки вероятности переобучения для пороговых конъюнкций

В работах К.В.Воронцова [8, 10, 57] предложен метод порождающих и запрещающих множеств, и с его помощью получены точные оценки вероятности переобучения для ряда модельных семейств алгоритмов. Этот метод основан на предположении, что для любого алгоритма $a \in A$ можно в явном виде записать необходимое и достаточное условие того, что он будет выбран методом обучения μ по выборке X. Для реальных семейств нахождение этих условий оказывается трудной задачей.

В данной главе предлагается упростить вид этих условий, ослабив их до необходимых, что приводит к верхним оценкам вероятности переобучения. Завышенность этих оценок оказывается относительно невелика, поскольку они существенно учитывают свойства расслоения и связности семейства алгоритмов. Как было показано в [54, 56], именно эти два свойства определяющим образом влияют на вероятность переобучения.

По сравнению с [57], данный подход не только существенно упрощает доказательства, но и приводит к общей оценке расслоения-связности. Оценки [57] опирались на сильное дополнительное предположение о существовании в семействе A корректного алгоритма, не допускающего ошибок на полной выборке $\mathbb{X}$. На практике данное предположение, как правило, не выполняется. В данной работе это избыточное предположение устранено.

Далее полученные верхние оценки вероятности переобучения применяются к семейству конъюнкций пороговых предикатов, предлагаются численные методы для вычисления этих оценок, приводятся результаты эмпирического анализа

завышенности оценок.

2.1 Метод порождающих и запрещающих множеств

Пусть $\mathbb{X} = \{x_1, \ldots, x_L\}$ и множество A состоит из алгоритмов a с попарно различными *векторами ошибок* $\big(I(a, x_i)\big)_{i=1}^{L}$.

Метод порождающих и запрещающих множеств [57], основан на гипотезе, что для любого алгоритма $a \in A$ можно записать необходимое и достаточное условие того, что он будет выбран методом обучения μ по выборке X. В простейшем случае это условие представляет собой требование, чтобы некоторое множество объектов $X_a \subset \mathbb{X}$, называемое *порождающим*, обязательно находилось в обучающей выборке, а второе множество объектов $X'_a \subset \mathbb{X}$, называемое *запрещающим*, обязательно находилось в контрольной выборке:

$$\big[\mu X{=}a\big] = \big[X_a \subseteq X\big]\big[X'_a \subseteq \bar{X}\big], \quad \forall X \subset \mathbb{X}\colon |X| = \ell. \tag{2.1}$$

Данное предположение приводит к точному выражению для вероятности того, что алгоритм $a \in A$ будет выбран методом обучения μ по выборке X.

$$P_a = \mathsf{P}\big[\mu X{=}a\big] = \frac{C_{L_a}^{\ell_a}}{C_L^{\ell}},$$

где

$$L_a = L - |X_a| - |X'_a|;$$
$$\ell_a = \ell - |X_a|;$$

и точной оценке вероятности переобучения:

$$Q_\varepsilon = \sum_{a \in A} P_a H_{L_a}^{\ell_a, m_a}\left(s_a(\varepsilon)\right), \tag{2.2}$$

где

$$m_a = n(a, \mathbb{X}) - n(a, X_a) - n(a, X'_a);$$
$$s_a(\varepsilon) = \tfrac{\ell}{L}\big(n(a, \mathbb{X}) - \varepsilon k\big) - n(a, X_a).$$

В общем случае одной парой множеств (X_a, X'_a) обойтись не удаётся, и приходится вводить множество пар (X_{av}, X'_{av}), $v \in V_a$:

$$\big[\mu X{=}a\big] = \sum_{v \in V_a} c_{av} \big[X_{av} \subseteq X\big] \big[X'_{av} \subseteq \bar{X}\big],$$

где $X_{av} \subset \mathbb{X}$ — *порождающие* множества, $X'_{av} \subset \mathbb{X}$ — *запрещающие* множества, c_{av} — числовые коэффициенты; обычно $c_{av} \in \{-1, +1\}$. Записанное в таком виде, условие $[\mu X{=}a]$ приводит к аналогичной оценке, которая отличается наличием дополнительного суммирования по всем индексным множествам V_a:

$$\begin{aligned}
Q_\varepsilon &= \sum_{a \in A} \sum_{v \in V_a} \frac{C_{L_{av}}^{\ell_{av}}}{C_L^\ell} H_{L_{av}}^{\ell_{av}, m_{av}} \left(s_{av}(\varepsilon)\right), \\
L_{av} &= L - |X_{av}| - |X'_{av}|; \\
\ell_{av} &= \ell - |X_{av}|; \\
m_{av} &= n(a, \mathbb{X}) - n(a, X_{av}) - n(a, X'_{av}); \\
s_{av}(\varepsilon) &= \tfrac{\ell}{L}\big(n(a, \mathbb{X}) - \varepsilon k\big) - n(a, X_{av}).
\end{aligned}$$

Доказано, что система порождающих и запрещающих множеств $(X_{av}, X'_{av})_{v \in V_a}$ может быть указана всегда, для любых $\mathbb{X}$, A и μ. Однако она может быть указана не единственным способом, и от выбора конкретной системы зависит вычислительная эффективность оценки. В худшем случае число операций, необходимых для вычисления оценки, имеет тот же порядок, что и число разбиений $O(C_L^\ell)$. Этот случай не представляет практического интереса, поскольку за такое число операций значение Q_ε можно вычислить непосредственно по определению. Очевидно, оценка тем эффективнее, чем меньше мощности множеств V_a, X_{av}, X'_{av}. Все известные точные оценки вероятности переобучения основаны на том, что модельные семейства алгоритмов обладают «регулярным» в том или ином смысле строением, что и позволяет эффективно формулировать условия $[\mu X{=}a]$ для всех $a \in A$ в терминах порождающих и запрещающих множеств.

В данной работе предлагается более простое необходимое условие, которое приводит к верхней оценке вероятности переобучения.

Гипотеза 2.1. Множество A, выборка $\mathbb{X}$ и метод μ таковы, что для каждого алгоритма $a \in A$ можно указать два множества: *порождающее* $X_a \subset \mathbb{X}$ и *запрещающее* $X'_a \subset \mathbb{X}$ такие, что

$$\big[\mu X{=}a\big] \leqslant \big[X_a \subseteq X\big]\big[X'_a \subseteq \bar{X}\big]. \tag{2.3}$$

Теорема 2.1. *Если гипотеза 2.1 верна, то для любого* $\varepsilon \in (0,1)$ *справедлива верхняя оценка вероятности переобучения*

$$Q_\varepsilon \leqslant \sum_{a \in A} P_a H_{L_a}^{\ell_a, m_a}\left(s_a(\varepsilon)\right); \tag{2.4}$$

$$\mathsf{P}[\mu X{=}a] \leqslant P_a = \frac{C_{L_a}^{\ell_a}}{C_L^\ell},$$

$$L_a = L - |X_a| - |X'_a|,$$

$$\ell_a = \ell - |X_a|,$$

$$m_a = n(a,\mathbb{X}) - n(a,X_a) - n(a,X'_a),$$

$$s_a(\varepsilon) = \tfrac{\ell}{L}\big(n(a,\mathbb{X}) - \varepsilon k\big) - n(a,X_a).$$

Доказательство аналогично приведённому в [57], с единственным отличием — равенства в (2.1) и (2.2) заменяются на неравенства в (2.3) и (2.4).

2.2 Оценки расслоения–связности

Определим отношение порядка на алгоритмах $a \leqslant b$ как естественное отношение порядка на их векторах ошибок: $I(a,x) \leqslant I(b,x)$ для всех $x \in \mathbb{X}$.

Определим метрику на алгоритмах как хэммингово расстояние между их векторами ошибок:

$$\rho(a,b) = \sum_{i=1}^{L} |I(a,x) - I(b,x)|.$$

Алгоритмы a и b называются *связанными*, если $a \leqslant b$ и $\rho(a,b) = 1$.

Будем полагать, что все векторы ошибок, порождаемые алгоритмами из A, попарно различны. Если это не так, то алгоритмы, соответствующие дублирующим векторам ошибок, исключим из множества A.

Определение 2.1. *Графом расслоения-связности множества алгоритмов A будем называть направленный граф $\langle A, E\rangle$, множество вершин которого совпадает с множеством алгоритмов, а множество рёбер E образуется всеми парами алгоритмов (a, b), такими, что $a \leqslant b$ и $\rho(a, b) = 1$.*

Граф расслоения–связности изоморфен графу введённого выше естественного отношения порядка на A (точнее, его минимальной транзитивной редукции) и является многодольным графом, в котором долями являются *слои алгоритмов* $A_m = \{a \in A \colon n(a, \mathbb{X}) = m\}$. Рёбрами могут соединяться только алгоритмы соседних слоёв.

Каждому ребру $(a, b) \in E$ графа расслоения–связности соответствует один и только один объект $x_{ab} \in \mathbb{X}$, такой, что $I(a, x_{ab}) = 0$ и $I(b, x_{ab}) = 1$.

Для каждого алгоритма $a \in A$ определим два множества объектов: X_a — множество объектов x_{ab}, соответствующих всевозможным рёбрам графа $(a, b) \in E$, исходящим из a:

$$X_a = \{x \in \mathbb{X} \mid \exists b \in A \colon I(a, x) < I(b, x),\ a \leqslant b,\ \rho(a, b) = 1\}, \tag{2.5}$$

и X'_a — множество объектов $x \in \mathbb{X}$, соответствующих рёбрам графа $(b, c) \in E$, принадлежащих всевозможным путям, ведущим в a:

$$X'_a = \{x \in \mathbb{X} \mid \exists b \in A \colon I(b, x) < I(a, x),\ b \leqslant a\}. \tag{2.6}$$

Лемма 2.2. *Если μ — метод пессимистичной минимизации эмпирического риска, то множества X_a (2.5) и X'_a (2.6) являются, соответственно, порождающим и запрещающим для алгоритма a в смысле Гипотезы 2.1.*

Доказательство. Докажем от противного, что если $\mu X = a$, то $X_a \subseteq X$ и $X'_a \subseteq \bar{X}$.

Допустим, что найдётся $x_{ab} \in X_a$, не лежащий в X. Тогда $n(a, X) = n(b, X)$, поскольку векторы ошибок a и b отличаются только на объекте x_{ab}. В то же время, $n(a, \mathbb{X}) + 1 = n(b, \mathbb{X})$, поэтому для метода μ, в силу его пессимистичности, выбор алгоритма b по выборке X будет предпочтительнее, чем a, что противоречит условию $\mu X = a$. Значит, $X_a \subseteq X$.

Допустим теперь, что найдётся $x \in X'_a$, лежащий в X. Тогда существует $b \in A$, для которого $n(b, X) < n(a, X)$. Поскольку метод μ минимизирует эмпирический риск, выбор алгоритма b будет предпочтительнее, чем a, что противоречит условию $\mu X = a$. Значит, $X'_a \subseteq \bar{X}$.

Лемма доказана. ■

Определение 2.2. *Связностью $q(a)$ алгоритма $a \in A$ будем называть число рёбер графа, исходящих из вершины a:*

$$q(a) = |X_a| = \#\big\{b \in A \;\big|\; (a, b) \in E\big\}.$$

Неформально говоря, связность $q(a)$ есть реализуемое алгоритмами семейства A число способов испортить алгоритм a так, чтобы он стал допускать на одну ошибку больше. Можно сказать и так, что это эффективное число свободных параметров семейства A в локальной окрестности алгоритма $a \in A$ относительно заданной выборки $\mathbb{X}$.

Определение 2.3. *Неоптимальностью $r(a)$ алгоритма $a \in A$ будем называть число объектов $x \in \mathbb{X}$, на которых алгоритм a ошибается, при том, что существует алгоритм $b \in A$, лучший, чем a (то есть $b \leqslant a$), не ошибающийся на x:*

$$r(a) = |X'_a| = \#\big\{x \in \mathbb{X} \;\big|\; \exists b \in A\colon I(b, x) < I(a, x),\ b \leqslant a\big\}.$$

Заметим, что неоптимальность $r(a)$ не обязательно равна числу ошибок на генеральной выборке $n(a, \mathbb{X})$. Равенство $r(a) = n(a, \mathbb{X})$ достигается в случае, когда существует корректный алгоритм $a_0 \in A$: $n(a_0, \mathbb{X}) = 0$. В общем случае можно утверждать лишь, что $r(a) \leqslant n(a, \mathbb{X})$. В графе расслоения-связности неоптимальность — это число различных объектов x_{bc}, соответствующих всевозможным рёбрам (b, c) на путях, ведущих к вершине a.

Теорема 2.3. *Если μ — метод пессимистичной минимизации эмпирического риска, то для вероятности переобучения справедлива оценка сверху:*

$$Q_\varepsilon(\mu, \mathbb{X}) \leqslant \sum_{a \in A} \frac{C^{\ell - q(a)}_{L - q(a) - r(a)}}{C^\ell_L} H^{\ell - q(a),\, n(a, \mathbb{X}) - r(a)}_{L - q(a) - r(a)} \left(\tfrac{\ell}{L}(n(a, \mathbb{X}) - \varepsilon k)\right). \tag{2.7}$$

Доказательство. Данная оценка следует непосредственно из общей оценки (2.4) и Леммы 2.2, если заметить, что произвольный алгоритм $a \in A$ ошибается на всех объектах запрещающего множества X'_a и не ошибается на всех объектах порождающего множества X_a: $|X_a| = q(a),\ n(a, X_a) = 0,\ |X'_a| = n(a, X'_a) = r(a)$. ■

Если в этой оценке положить $q(a) = r(a) = 0$ для всех a, то получится оценка Вапника-Червоненкиса:

$$Q_\varepsilon(\mu, \mathbb{X}) \leqslant \sum_{a \in A} H_L^{\ell, n(a,\mathbb{X})}\left(\tfrac{\ell}{L}(n(a,\mathbb{X}) - \varepsilon k)\right) \leqslant$$
$$\leqslant |A| \cdot \max_{m=0,\ldots,L} H_L^{\ell,m}\left(\tfrac{\ell}{L}(m - \varepsilon k)\right) \overset{\text{при } \ell=k}{\leqslant} |A| \cdot e^{-\varepsilon^2 \ell}.$$

Оценка Вапника-Червоненкиса является сильно завышенной, поскольку она не учитывает свойства расслоения и связности семейства алгоритмов [54, 55, 56].

2.3 Структура классов эквивалентности семейства пороговых конъюнкций

Оценки вероятности переобучения, полученные выше для алгоритмов классификации, легко переносятся на логические правила, если соответствующим образом определить индикатор ошибки: $I(r, x_i) = \big[r(x_i) \neq [y_i{=}y]\big]$. Аналогично вводятся понятия *связности* правил, *отношение порядка* и *расстояние* между правилами.

Без ограничения общности будем полагать, что в правилах вида (1.3) множество признаков ω фиксировано, и все знаки сравнения суть $\leqslant$. В общем случае в пороговых предикатах возможны $2^n C_n^r$ вариантов сочетания $r = |\omega|$ признаков и знаков $\leqslant, \geqslant$. Предполагается, что перебор этих вариантов осуществляется некоторым стандартным алгоритмом поиска информативных правил, причём не важно, является ли перебор полным, или это некоторый сокращённый эвристический поиск. Наша задача заключается в том, чтобы оценить вероятность переобучения, возникающего в результате оптимизации порогов c_j по обучающей

выборке при фиксированном наборе признаков ω и фиксированных знаках неравенств $\leqslant$.

Допустим, что значения x_i^j каждого признака $j \in \omega$ попарно различны на объектах $x_i \in \mathbb{X}$. Тогда без ограничения общности можно предполагать, что все признаки принимают целые значения $1, \ldots, L$, и никакие два объекта не имеют равных значений одного признака. Значения порогов c^j в правилах (1.3) можно выбирать также только из целых значений $0, \ldots, L$.

Пусть u — произвольный объект из $\mathbb{X}$. Будем говорить, что объект u *доминируется* множеством объектов $S \subseteq \mathbb{X}$ и писать $u \prec S$, если для каждого $j \in \omega$ существует объект $s \in S$, такой, что $u^j \leqslant s^j$. Если множество S состоит из одного элемента s и $u \prec S$, то будем записывать $u \prec s$.

Определение 2.4. *Подмножество $S \subseteq \mathbb{X}$ называется недоминирующимся, если любой объект $s \in S$ не доминируется подмножеством $S \backslash s$.*

Введём множество всех недоминирующихся подмножеств мощности q:

$$M_q = \left\{ S \subseteq \mathbb{X} \colon |S| = q,\ \forall s \in S \ \ s \not\prec S \backslash s \right\}.$$

Введем искусственное недоминирующееся подмножество S_0, состоящее из одного объекта $x_0 = (0, \ldots, 0)$. Обозначим $M_0 = \{S_0\}$. Правило $r(x; 0, \ldots, 0)$ будем обозначать r_0.

Перечислим некоторые полезные свойства недоминирующихся подмножеств.

1. Мощность любого недоминирующегося подмножества $|S|$ не превышает n.

2. Если подмножество S недоминирующееся, то любое его подмножество $S' \subset S$ также недоминирующееся.

3. $|M_q| \leqslant C_L^q$.

4. Для построения всех $S \in M_q$ достаточно добавить к каждому подмножеству $S' \in M_{q-1}$ один объект, еще не входящий в него; если полученное подмножество недоминирующееся, то оно войдет в M_q.

Лемма 2.4. *Для любого объекта x из недоминирующегося подмножества S*

найдется хотя бы один признак $j \in \omega$, для которого $x^j = \max\limits_{s \in S} s^j$, причём

$$\bigcup_{j=1}^{n} \operatorname{Arg}\max_{s \in S}(s^j) = S.$$

Доказательство. Допустим, что это не так. Тогда существует $x \in S$, такой, что для любого $j \in \omega$ существует $s \in S \backslash x$, для которого $x^j < s^j$. Следовательно $x \prec S \backslash x$, значит, S не является недоминирующимся подмножеством. ■

Поставим в соответствие подмножеству S правило

$$r(x, S) = r(x; \max_{x \in S} x^1, \ldots, \max_{x \in S} x^n).$$

Очевидно, разным S ставятся в соответствие разные $r(x, S)$, т. к. значения каждого признака x_i^j, $i = 1, \ldots, L$ попарно различны. Следовательно, зная j и x_i^j, можно однозначно указать объект x_i. Следовательно, набор параметров $c^j = \max\limits_{x \in S} x^j$, $j \in \omega$ задаёт подмножество S однозначно, и различным S не могут соответствовать одинаковые $r(x, S)$.

Функция индикатора ошибки индуцирует на множестве правил R *классы эквивалентности*. Два правила эквивалентны, $r \sim r'$, если их векторы ошибок совпадают.

На рис. 2.1 показан пример задачи с $n = 2$ признаками, $L = 10$ объектами и семейство R правил вида

$$r(x; c^1, c^2) = [x^1 \leqslant c^1]\,[x^2 \leqslant c^2].$$

Каждое правило из R задается парой порогов (c^1, c^2), поэтому правилам соответствуют узлы прямоугольной сетки $\{0, \ldots, L\}^2$. Отрезками соединены правила, лежащие в одном классе эквивалентности. Рядом с каждым классом эквивалентности подписано число ошибок на полной выборке $n(r, \mathbb{X})$. Все одноэлементные подмножества объектов являются недоминирующимися. Среди двухэлементных подмножеств недоминирующимися являются только пары несравнимых объектов, например, пара объектов $\big\{(2, 5), (5, 1)\big\}$. Недоминирующихся подмножеств мощности выше 2 нет, т. к. размерность задачи $n = 2$.

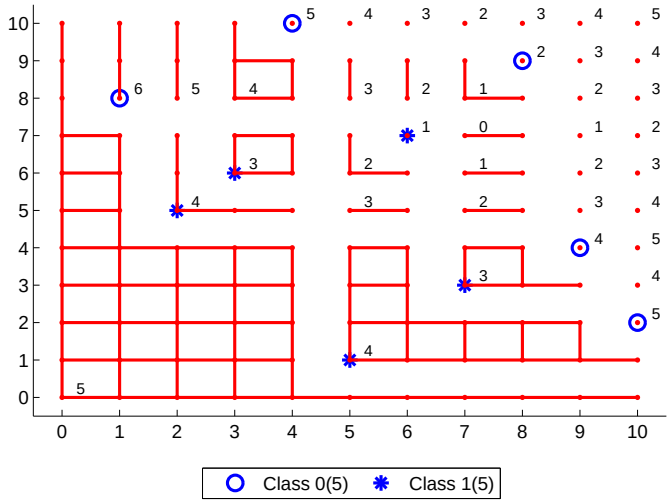

Рис. 2.1: Двумерная выборка длины $L = 10$, по 5 объектов в каждом классе (отмечены крупными точками).

Лемма 2.5. *Пусть $E \subset R$ — класс эквивалентности правил, $c^j(r)$ — j-й параметр правила r. Классу E принадлежит правило*

$$r_E(x) = r\big(x; \min_{r' \in E} c^1(r'), \ldots, \min_{r' \in E} c^n(r')\big).$$

Доказательство. В силу эквивалентности и бинарности всех правил $r' \in E$ правило

$$r(x) = \prod_{r' \in E} r'\big(x; c^1(r'), \ldots, c^n(r')\big)$$

принимает на всех объектах $x \in \mathbb{X}$ те же значения, $r(x) = r'(x)$, что и любое правило r' из E. Кроме того, r представимо в виде (1.3):

$$\begin{aligned} r(x) &= \prod_{r' \in E} \prod_{j \in \omega} \big[x^j \leqslant c^j(r')\big] = \\ &= \prod_{j \in \omega} \big[x^j \leqslant \min_{r' \in E} c^j(r')\big] = r_E(x). \end{aligned}$$

Таким образом, правило $r_E(x)$ также принадлежит E. Что и требовалось доказать. ■

Будем называть правило $r_E(x)$ *стандартным представителем* класса эквивалентности E. На рис. 2.1 стандартные представители соответствуют левым нижним точкам каждого класса эквивалентности: $(0,0)$, $(1,8)$, $(2,5)$, $(5,1)$, и т. д.

Возможны и другие графические представления структуры классов эквивалентности.

На рис. 2.2 изображен граф связей между стандартными представителями классов эквивалентности правил. Каждая вершина этого графа — это стандартный представитель класса эквивалентности, а связь между вершинами означает, что векторы ошибок правил различаются на одном объекте. Заметим, что граф зависит только от значений признаков объектов, но не зависит от классификации объектов.

Если добавить классификацию объектов, то каждому классу эквивалентности E, можно поставить в соответствие число ошибок $n(r_E, \mathbb{X})$. Числа ошибок указаны на рис. 2.2 рядом с точками — стандартными представителями классов эквивалентности. Граф расслоения-связности данного семейства правил изоморфен графу связей и является многодольным графом, каждый слой (доля) которого образуется правилами с равными значениями $n(r_E, \mathbb{X})$, рис. 2.3. Слои располагаются в порядке возрастания числа ошибок снизу вверх. В данном примере есть корректное правило, не допускающее ошибок на полной выборке.

Теорема 2.6. *Существует взаимно однозначное соответствие между множеством всех классов эквивалентности и множеством всех недоминирующихся подмножеств.*

Доказательство. Построим по заданному стандартному представителю $r_E(x)$ недоминирующееся подмножество S. Если $r_E(x) = r_0$, то $S = S_0$. Иначе пусть $r_E(x) = r(x; c^1, \ldots, c^n)$. Рассмотрим множество объектов $U = \{x\colon r_E(x) = 1\}$. Оно не пусто, так как в противном случае класс эквивалентности E содержал бы r_0. Выберем из множества U недоминирующееся подмножество

$$S = \bigcup_{j=1}^{n} \operatorname{Arg}\max_{x \in U}(x^j).$$

Пусть $r(x,S) = r(x;d^1,\ldots,d^n)$. Докажем, что $r_E(x) = r(x,S)$ от противного. Допустим, что некоторые пороги в правилах $r_E(x)$ и $r(x,S)$ не совпадают; скажем, для определенности, $d^1 < c^1$. Возьмем правило $r(x;d^1,c^2,\ldots,c^n)$. Оно принадлежит E, так как $d^1 = \max\limits_{x\in U} x^1$, и при фиксированных параметрах $c^2,\ldots,c^n$ нет объекта, на котором, при увеличении первого порога от d^1 до c^1, изменялся бы вектор ошибок. Следовательно, $r_E(x)$ не может быть стандартным представителем класса эквивалентности E. Получили противоречие.

Теперь докажем обратное: любому недоминирующемуся подмножеству объектов S соответствует один и только один класс эквивалентности E. Поставим в соответствие недоминирующемуся подмножеству S класс эквивалентности E, такой, что $r(x,S) \in E$. Покажем от противного, что не существует второго такого S', $S' \neq S$, что $r(x,S') \in E$. Пусть это не так: $r(x,S) = r(x;c^1,\ldots,c^n) \in E$, $r(x,S') = r(x;d^1,\ldots,d^n) \in E$ и, для определенности, $d^1 < c^1$. Но тогда существует объект $x_* = \arg\max\limits_{x\in S}(x^1)$, такой, что $x_*^1 = c^1$. При этом $r(x_*,S) = 1$ и $r(x_*,S') = 0$, следовательно эти два правила не могут принадлежать одному классу эквивалентности. ■

Следствие 2.1. *Для каждого класса эквивалентности E существует единственное недоминирующееся подмножество S, такое, что $r_E(x) = r(x,S)$.*

Следствие 2.2. *Число классов эквивалентности равно* $\sum\limits_{q=0}^{n} |M_q|$.

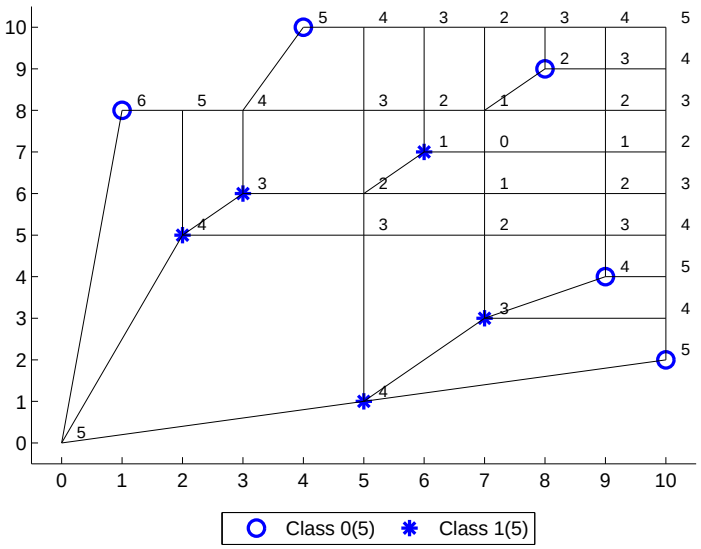

Рис. 2.2: Граф связей между стандартными представителями классов эквивалентности правил для выборки, показанной на рис. 2.1.

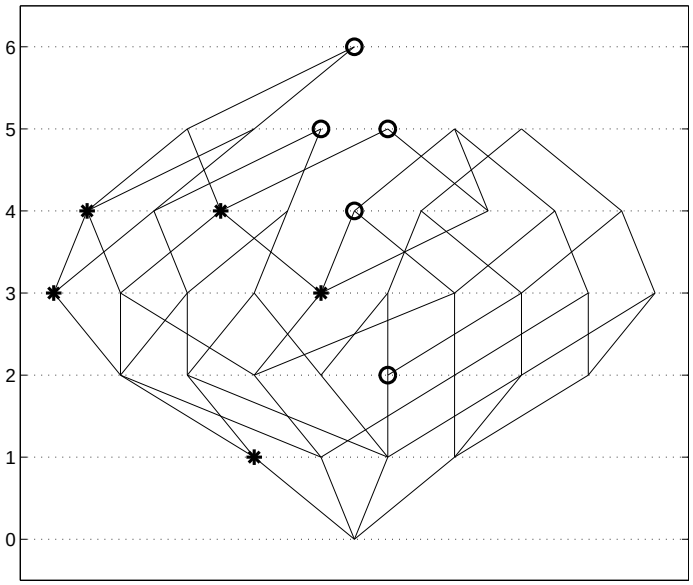

Рис. 2.3: Граф расслоения–связности, изоморфный графу связей на рис. 2.2. По вертикальной оси отложено число ошибок $n(r, \mathbb{X})$.

2.4 Обобщение на случай номинальных и порядковых признаков

Полученные результаты обобщаются на случай, когда числовой признак может принимать одинаковые значения на различных объектах, а также на порядковые и номинальные признаки.

Порядковые признаки. До этого момента мы рассматривали признаки каждый объект на котором, имел уникальное значение признака. Это предположение будет выполнено с вероятностью 1, если значения признака выбирались из непрерывного распределения на $\mathbb{R}$. Однако, в реальных задачах часто встречаются признаки для которых число значений ограничено, и поэтому с ростом выборки значения этого признака для разных объектов начнут повторяться. Примером такого признака может являться возраст человека измеренный в годах.

После снятия ограничения уникальности значений признаков, может оказаться что в выборке $\mathbb{X}$ существуют два объекта u и v, такие что у них полностью совпадают значения на всех признаках: $u^j = v^j$, для всех $j \in \omega$. При построении множества недоминирующихся подмножеств будем рассматривать сокращенную выборку $\mathbb{X}$, в которую входит только один объект из каждого подмножества объектов с полностью совпадающими значениями признаков.

Пусть x — произвольный объект из недомнинирующегося подмножества S, будем обозначать $Ind_S(x) \subseteq \omega$ — индексы тех признаков, для которых $x^j = \max\limits_{s \in S} s^j$.

Дадим более общее определение недоминирующегося подмножества объектов.

Определение 2.5. *Подмножество $S \subseteq \mathbb{X}$ называется недоминирующимся, если любой объект $s \in S$ не доминируется подмножеством $S \backslash s$, и не существует такого объекта $x \in \mathbb{X} \backslash S$, что $Ind_S(s) = Ind_{x \cup S \backslash s}(x)$.*

Такое обобщение понятия позволяет однозначно определять по правилу r недоминирующееся подмножество S такое, что $r = r(x, S)$.

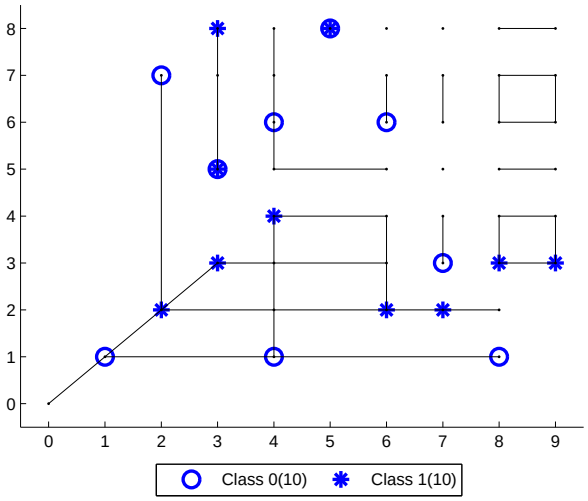

Рис. 2.4: Граф связей для выборки $L = 20$ и $n = 2$. Оба признака упорядоченные с повторяющимися значениями.

Для упорядоченных признаков с повторяющимися значениями число компонент связности в графе связей может быть больше единицы, пример на рис. 2.4.

Номинальные признаки. Область значений номинальных признаков это конечное множество. Условия на номинальных признаках являются равенствами $[x^j = a]$.

Теорема 2.7. *Пусть правила*

$$r(x) = [x^t = a] \prod_{j \in \omega \setminus t} \left[x^j \leqslant_j c_1^j\right] = [x^t = a][r'(x) = 1]$$

$$q(x) = [x^t = b] \prod_{j \in \omega \setminus t} \left[x^j \leqslant_j c_2^j\right] = [x^t = b][q'(x) = 1]$$

различаются только одним термом на t-том признаке, выделяют объекты класса y и отличны от r_0. Тогда правила r и q — несвязаны.

Доказательство. Если ни одно правило не является r_0, то существуют объекты x_r, x_q такие, что $r(x_r) = 1,\ q(x_q) = 1$. Следовательно $x_r^t = a,\ x_q^t = b;\ x_r \neq x_q$; $r(x_q) = 0, q(x_r) = 0$. Допустим что правила r и q связаны, для определенности,

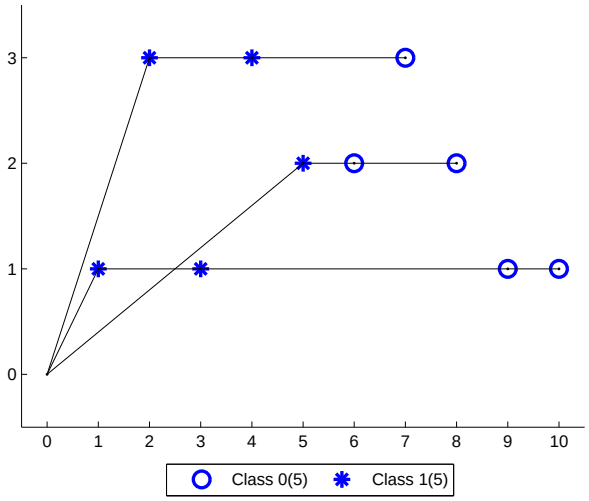

Рис. 2.5: Граф связей для выборки $L = 10$ и $n = 2$. Первый признак (по оси абсцис) упорядоченный, второй (по оси ординат) — номинальный с 3 значениями.

$r < q$, тогда существует единственный объект x_* такой, что

$$0 = \big[[r(x_*) = 1] \neq [y_* = y]\big] < \big[[q(x_*) = 1] \neq [y_* = y]\big] = 1,$$

и для всех остальных объектов $x \in \mathbb{X} \backslash x_*$:

$$\big[[r(x_*) = 1] \neq [y_* = y]\big] = \big[[q(x_*) = 1] \neq [y_* = y]\big].$$

Пусть $y_* = y$, тогда из правой части неравенства следует $x_* = x_r$, но

$$\big[[r(x_q) = 1] \neq [y_* = y]\big] = \big[0 \neq 1\big] = 1$$

$$\neq$$

$$\big[[q(x_q) = 1] \neq [y_* = y]\big] = \big[1 \neq 1\big] = 0.$$

Противоречие. Аналогично и для $y_* \neq y$. ■

Пример графа связей для номинального признака изображен на рис. 2.5.

Следствие 2.3. *Выборку можно разделить на группы объектов по сочетанию значений номинальных признаков. Рассматривая каждую группу в отдельности получим задачу оценки вероятности переобучения для упорядоченных и числовых признаков.*

2.5 Численные методы оценивания вероятности переобучения

Рассмотрим задачу вычисления верхней оценки вероятности переобучения по формуле (2.7) для семейства правил вида (1.3).

Предлагается производить суммирование по классам эквивалентности правил, перебирая их по слоям, от нижнего слоя к верхнему. Преимущество этого подхода в том, что вклады слоёв в оценку $\bar{Q}_\varepsilon$ быстро убывают с ростом номера слоя. Небольшого числа нижних слоёв может оказаться достаточно для получения оценки приемлемой точности. Тогда верхними слоями можно будет пренебречь, и тем самым существенно сократить объём вычислений.

Для перебора классов эквивалентности будем перебирать недоминирующиеся подмножества объектов. Начнём с того, что найдём правило с минимальным числом ошибок. Затем, для перехода со слоя m на слой $m+1$ будем перебирать для каждого правила r из m-го слоя все правила, связанные с ним (назовём множество таких правил окрестностью правила r), и из них брать только правила, допускающие на одну ошибку больше, то есть лежащие в слое $m+1$. Единожды отобранные правила будем помечать, чтобы не отобрать их ещё раз при просмотре окрестности другого правила r' из слоя m.

Рассмотрим подзадачу нахождения всех правил, связанных с данным. Пусть правило r принадлежит классу эквивалентности E, и ему соответствует стандартный представитель r_E и набор недоминирующихся объектов S. Каждому правилу соответствует n порогов $c^1, \ldots, c^n$, а каждому объекту x_i соответствует n значений признаков $x_i^1, \ldots, x_i^n$. Будем рассматривать их как элементы одного дискретного пространства — n-мерной сетки $\{0, \ldots, L\}^n$.

Обобщим понятие доминируемости на отношения объектов и правил. Будем говорит что правило r доминирует (покрывает) объект x, если $x^i \leqslant c^i,\ i = 1, \ldots, n$.

Все объекты выборки $\mathbb{X}$ можно разделить на четыре типа по отношению

к правилу r:

- S_r — объекты недоминирующегося множества, соответствующего правилу r;
- $S_r^h = \{x | r \prec x\}$ — объекты, которые доминируют правило r;
- $S_r^l = \{x | x \prec r, \quad x \notin S_r\}$ — объекты, которые доминируются правилом r;
- $S_r^n = X \backslash (S_r \cup S_r^h \cup S_r^l)$ — объекты, несравнимые с правилом r.

По определению правила r и q называются связанными, если $\rho(r, q) = 1$ и $r \leqslant q$. Существует единственный объект $x_{rq} \in \mathbb{X}$ такой, что $0 = I(r, x_{rq}) < I(q, x_{rq}) = 1$.

Для работы данного алгоритма нам потребуется предвычисленная информация об отношениях доминирования между объектами выборки.

Построим минимальную транзитивную редукцию графа отношений объектов в выборке $G = \langle \mathbb{X}, E \rangle$. Вершинами графа G будут объекты множества $\mathbb{X}$, ребро $(u, v) \in E$ между вершинами $u, v \in \mathbb{X}$ существует тогда и только тогда, когда $u < v$ и не существует объекта $x \in \mathbb{X}$ такого, что $u < x < v$. Для построения графа G сначала воспользуемся топологической сортировкой, описанной в [23]. После этого, перебирая объекты по-порядку удалим лишние связи. Функция $g(x)$ выдает по заданному объекту x все объекты

Допустим что мы хотим найти все правила связанные с правилом r, тогда нам надо придерживаться алгоритма 2.5.1.

Шаг 0. Подготовительный. Первое что нужно сделать это определить для данного правила r множества S_r, S_r^h, S_r^l, S_r^n. Сложность этой операции $O(nL)$. В результате мы можем дальше рассматривать в качестве правила, стандартного представителя класса эквивалентности, к которому принадлежит правило r. Стандартный представитель однозначно определяется по множеству S_r за время $O(n^2)$.

Шаг 1. Связанные правила доминируемые данным. Эти правила получаются исключением из недоминирующегося подмножества объектов S_r соответствующего правилу, поочередно всех объектов. Таким образом уменьшается на один число покрытых правилом r объектов, а значит и вектор ошибок изменяется на одном объекте. Число связанных правил получаемых на данном шаге, равно в точности числу объектов в недоминирующемся подмножестве S_r. Исключенный объект x, по лемме 2.4, достигает максимума среди объектов из S_r на некоторых признаках ($h = Ind_{S_r}(x)$ — индексы этих признаков). Для каждого признака из h, найдем среди объектов из доминируемых правилом r множества S_r^l такой, что является максимальным по этому признаку. Добавив все найденные объекты к $S_r \backslash x$, мы получим новое недоминирующееся подмножество. Соответствующее данному подмножеству правило будет связано с r. Сложность этого шага $O(nL)$.

Шаг 2. Связанные правила доминирующие данное. Такие правила могут быть образованы только недоминирующимися множествами из одного объекта $x \in S_r^h$. Мы добавляем объект x к покрытым правилом r. Множество объектов значений функции $g(x)$ должно принадлежать множеству $S_r \cup S_r^l$. Только в этом случае векторы ошибок правила r и искомого правила, будут отличаться на одном объекте x. Сложность этого шага $O(nL)$.

Шаг 3. Связные правила несравнимые с данным. Такие правила могут быть образованы только недоминирующимися множествами, содержащими объекты из S_r^n. Объект x, который дополнительно покроет правило q, связанное с r, должен обладать следующими свойствами: $g(x) \subseteq S_r \cup S_r^l$, не должно существовать объекта $u \neq x$ такого, что $q(u) = 1$ и $r(u) = 0$. Сложность этого шага в наихудшем случае $O(nL^2)$.

Общая сложность всех шагов алгоритма 2.5.1, в худшем случае, составляет $O(nL^2)$. Однако в среднем ее можно улучшить если активно использовать предвычисленные данные графа G.

Для эффективного вычисления оценки вероятности переобучения по лемме 2.2

предлагается послойный алгоритм перебора классов эквивалентности. Вклады слоёв в вероятность переобучения быстро убывают с ростом номера слоя, равного числу ошибок правила $n(r, \mathbb{X})$. Поэтому для вычисления оценки достаточно обойти лишь несколько нижних слоёв, содержащих небольшую долю всех классов эквивалентности, и тем самым избежать перебора основной массы классов эквивалентности.

Пример 2.1. Следует отметить, что множество X_a не является максимально возможным, то есть для некоторых алгоритмов a его можно расширить. Чем полнее будут указаны множества X_a и X'_a для каждого алгоритма, тем точнее будет оценка вероятности переобучения.

Для выборок с небольшим числом объектов L, можно воспользоваться полным перебором всех возможных разбиений выборки $\mathbb{X} = X \sqcup \bar{X}$ на обучение и контроль. Для каждого алгоритма a найдем всевозможные разбиения выборки такие, что алгоритм выбирается на этом разбиении пессимистичным методом МЭР. Обозначим D_a получившееся множество разбиений выборки $\mathbb{X}$.

$$X_a = \bigcap_{X \sqcup \bar{X} \in D_a} X, \quad X'_a = \bigcap_{X \sqcup \bar{X} \in D_a} \bar{X}$$

Перечисление объектов которые попадают в X_a и X'_a для выборки изображенной на рисунке 2.1 представлено в таблице 2.5. Жирным выделены те объекты которые находятся полным перебором всех разбиений, но не находятся по лемме 2.2.

Алгоритм 2.5.1 Алгоритм получения правил, связанных с данным.

Вход:

r — рассматриваемое правило, связанные с которым надо найти;

S_r — недоминирующееся множество объектов соответствующее правилу r;

S_r^l — доминирующееся правилом r множество;

S_r^h — множество объектов доминирующее правило r;

S_r^n — множество объектов несравнимых с правилом r;

$g(x)$ — функция, возвращающая список объектов, доминируемых объектом x;

Выход:

R_{Neib} — список связанных правил доминируемых правилом r.

для всех объектов $x \in S_r$

  $h = Ind_{S_r}(x)$ — индексы признаков по которым объект x доминирует в S_r;

  $S' = \bigcup\limits_{j \in h} \operatorname{Arg\,max}\limits_{u \in S_r^l}(u^j)$;

  $S = S_r \backslash x \cup S'$;

  $R_{Neib} = R_{Neib} \cup r(x, S)$;

для всех объектов $x \in S_r^h$

  если $g(x) \subseteq S_r \cup S_r^l$ **то**

    $R_{Neib} = R_{Neib} \cup r(\{x\})$;

для всех объектов $x \in S_r^n$

  если $g(x) \subseteq S_r \cup S_r^l$ **то**

    $S_r' = \{s \in S_r : s \not\prec x\} \cup x$;

    $q = r(S_r')$ — правило полученное добавлением в недоминирующееся множество объекта x;

    если не существует $u \neq x$ таких что $q(u) = 1, r(u) = 0$ **то**

      $R_{Neib} = R_{Neib} \cup q$;

Метод поиска Номер алгоритма	Лемма 2.2 X_a	Лемма 2.2 X'_a	Полный перебор X_a	Полный перебор X'_a
1	$\{1,4,8,9,10\}$	$\{2,3,5,6,7\}$	$\{1,4,8,9,10\}$	$\{2,3,5,6,7\}$
2	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$
3	$\{1,2\}$	$\{3,5,6,7\}$	$\{1,2\}$	$\{3,5,6,7\}$
4	$\{2,4,8,9,10\}$	$\{1,3,5,6,7\}$	$\{2,4,8,9,10\}$	$\{1,3,5,6,7\}$
5	$\{1,3\}$	$\{5,6,7\}$	$\{1,3\}$	$\{5,6,7\}$
6	$\{3,4\}$	$\{1,5,6,7\}$	$\{3,4\}$	$\{1,5,6,7\}$
7	$\{2,3,8,9,10\}$	$\{1,4,5,6,7\}$	$\{2,3,8,9,10\}$	$\{1,4,5,6,7\}$
8	$\{5,10\}$	$\{2,3,6,7\}$	$\{5,\mathbf{9},10\}$	$\{2,3,6,7\}$
9	$\{2,5\}$	$\{3,6,7\}$	$\{\mathbf{1},2,5,\mathbf{9}\}$	$\{3,6,7\}$
10	$\{1,3,5\}$	$\{6,7\}$	$\{1,3,5,\mathbf{9}\}$	$\{6,7\}$
11	$\{4,5\}$	$\{1,6,7\}$	$\{4,5\}$	$\{1,6,7\}$
12	$\{5\}$	$\{1,4,6,7\}$	$\{5\}$	$\{1,4,6,7\}$
13	$\{1,6\}$	$\{7\}$	$\{1,6,\mathbf{9}\}$	$\{7\}$
14	$\{4,6\}$	$\{1,7\}$	$\{4,6,\mathbf{8},\mathbf{9}\}$	$\{1,7\}$
15	$\{6\}$	$\{1,4,7\}$	$\{6,\mathbf{8}\}$	$\{1,4,7\}$
16	$\{7,9,10\}$	$\{2,3,6\}$	$\{7,9,10\}$	$\{2,3,6\}$
17	$\{2,7,9\}$	$\{3,6\}$	$\{2,7,9\}$	$\{3,6\}$
18	$\{3,7,9\}$	$\{6\}$	$\{\mathbf{1},3,7,9\}$	$\{6\}$
19	$\{1,6,7,9\}$	$\varnothing$	$\{1,6,7,9\}$	$\varnothing$
20	$\{4,7,8,9\}$	$\{1\}$	$\{4,7,8,9\}$	$\{1,\mathbf{2},\mathbf{5},\mathbf{10}\}$
21	$\{7,8\}$	$\{1,4\}$	$\{7,8,\mathbf{9}\}$	$\{1,4\}$
22	$\{4,9\}$	$\{1,8\}$	$\{4,9\}$	$\{1,8\}$
23	$\{9\}$	$\{1,4,8\}$	$\{9\}$	$\{1,4,8\}$
24	$\{10\}$	$\{2,3,6,9\}$	$\{10\}$	$\{2,3,6,9\}$
25	$\{2,10\}$	$\{3,6,9\}$	$\{2,10\}$	$\{3,6,9\}$
26	$\{3,10\}$	$\{6,9\}$	$\{\mathbf{1},3,10\}$	$\{6,9\}$
27	$\{1,6,10\}$	$\{9\}$	$\{1,6,10\}$	$\{9\}$
28	$\{8,10\}$	$\{1,9\}$	$\{8,10\}$	$\{1,9\}$
29	$\{4,10\}$	$\{1,8,9\}$	$\{4,10\}$	$\{1,8,9\}$
30	$\{10\}$	$\{1,4,8,9\}$	$\{10\}$	$\{1,4,8,9\}$
31	$\{1,4,5,8,9\}$	$\{2,3,6,7,10\}$	$\{1,4,5,8,9\}$	$\{2,3,6,7,10\}$
32	$\{7,9\}$	$\{2,3,6,10\}$	$\{7,9\}$	$\{2,3,6,10\}$
33	$\{1,4,5,7,8\}$	$\{2,3,6,9,10\}$	$\{1,4,5,7,8\}$	$\{2,3,6,9,10\}$
34	$\{2\}$	$\{3,6,9,10\}$	$\{2\}$	$\{3,6,9,10\}$
35	$\{3\}$	$\{6,9,10\}$	$\{1,3\}$	$\{6,9,10\}$
36	$\{1,6\}$	$\{9,10\}$	$\{1,6\}$	$\{9,10\}$
37	$\{8\}$	$\{1,9,10\}$	$\{8\}$	$\{1,9,10\}$
38	$\{4\}$	$\{1,8,9,10\}$	$\{4\}$	$\{1,8,9,10\}$
39	$\{2,3,5,6,7\}$	$\{1,4,8,9,10\}$	$\{2,3,5,6,7\}$	$\{1,4,8,9,10\}$

Таблица 2.1: Результаты поиска объектов X_a и X'_a.

2.6 Информативность пороговых конъюнкций с поправкой на переобучение

Рассмотрим правило r которое выделяет объекты класса y. Кроме индикатора полной ошибки $I(r, x_i) = \big[[r(x_i) = 1] \neq [y_i = y]\big]$ можно выделить еще два индикатора:

- $I_p(r, x_i) = \big[[r(x_i) = 0][y_i = y]\big]$ — число непокрытых «своих» объектов;
- $I_n(r, x_i) = \big[[r(x_i) = 1][y_i \neq y]\big]$ — число покрытых «чужих» объектов.

Введем отношения эквивалентности двух привил r и r' класса y для разных функций индикатора ошибки.

- $r \overset{p}{\sim} r'$ при $I_p(r, x_i)$;
- $r \overset{n}{\sim} r'$ при $I_n(r, x_i)$;
- $r \overset{err}{\sim} r'$ при $I(r, x_i) = \big[[r(x_i) = 1] \neq [y_i = y]\big] = I_p(r, x_i) + I_n(r, x_i)$.

Аналогично, можно ввести понятие эквивалентности по функционалу информативности: $r \overset{H}{\sim} r'$ при $H_{(P,N)}(p, n) = H_{(P,N)}(p', n')$.

Утверждение 2.8. *Если правило* $r \overset{err}{\sim} r'$, *то* $r \overset{H}{\sim} r'$, *где* $H_{(P,N)}(p, n)$ — *функционал информативности,* $P = |\{i\colon y_i = y\}|$ — *число объектов класса* y, $N = |\{i\colon y_i \neq y\}|$ — *число объектов других классов.*

Стандартные критерии информативности строятся исключительно по обучающей выборке и не учитывают эффект переобучения. При фиксированном подмножестве признаков ω переобучение возникает вследствие оптимизации порогов c_j, $j \in \omega$. Оно проявляется в том, что значения критериев $P(r, X)$ и $N(r, X)$ на обучающей выборке X оказываются оптимистично смещёнными относительно $P(r, \bar{X})$ и $N(r, \bar{X})$ на контрольной выборке $\bar{X}$. Предположим, что оценка $P(r, X)$ завышена на ΔP, оценка $N(r, X)$ занижена на ΔN. Предлагается с помощью

оценок вероятности переобучения, и затем обращения этих оценок, оценить величину смещений ΔP, ΔN, чтобы при отборе закономерностей пользоваться модифицированным критерием информативности $H(P - \Delta P, N + \Delta N)$. Величина смещений ΔP, ΔN нетривиальным образом зависит от выборки, поэтому введение таких поправок может существенно влиять на отбор закономерностей во множества R_y.

2.7 Численные эксперименты на модельных данных

2.7.1 Анализ завышенности оценки расслоения-связности

Будем генерировать модельные выборки данных со следующими характеристиками: число признаков $n = 2$, число классов $Y = \{0, 1\}$, число объектов $L = 100$, по 50 объектов в каждом классе. Сгенерируем четыре модельные выборки: «Correct», «Noise10», «Noise20» и «Random», отличающиеся только классификацией объектов. Для выборки «Correct» существует правило, разделяющее два класса без ошибок. Выборка «Noise10» получается из «Correct» небольшим зашумлением: для 10 пограничных объектов класс меняется на противоположный. Для выборки «Noise20» класс меняется у 20 объектов. Выборка «Random» получается случайным назначением классов всем объектам. В качестве ориентира используем оценки вероятности переобучения $\hat{Q}_\varepsilon$, вычисляемые методом Монте-Карло по 100 случайным разбиениям $\mathbb{X} = X \sqcup \bar{X}$ на обучение X и контроль $\bar{X}$, см. рис. 2.6.

Завышенность оценки (2.4) относительно невелика и возрастает с уровнем шума, см. рис. 2.7, 2.8. Оценка Вапника-Червоненкиса не зависит от данных, одинакова для всех четырёх выборок и сильно завышена, рис. 2.9.

Для тех же четырёх модельных выборок на рис. 2.10 показаны зависимости величины завышенности $Q_\varepsilon / \hat{Q}_\varepsilon$ оценки (2.4) от ε. Эта зависимость неустойчива. Тем не менее, в диапазоне значений $\varepsilon = 0.01 \div 0.1$, представляющих наибольший

практический интерес, величина завышенности примерно одинакова для всех выборок и имеет порядок $20 \div 30$.

На рис. 2.11 по горизонтальной оси откладывается значение числа ошибок m правил на полной выборке. Кривая «Count» — это распределение правил по числу ошибок; для неё по вертикальной оси откладывается доля правил с m ошибками. Кривая «Vklad» — это распределение правил по их относительному вкладу в вероятность переобучения; для неё по вертикальной оси откладывается отношение $\hat{Q}(m)/\hat{Q}$, где $\hat{Q}(m)$ — суммарный вклад правил с m ошибками в функционал $\hat{Q}$. Графики построены для тех же выборок «Correct», «Noise10», «Noise20», «Random». В случае корректной выборки «хорошие» правила существенно чаще выбираются методом обучения, и именно они вносят основной вклад в вероятность переобучения. Чем выше уровень шума, тем сильнее сближаются кривые «Count» и «Vklad».

2.7.2 Экспериментальное подтверждение методики

Эксперименты, описанные в данном разделе, проводились на двух модельных выборках длины $L = 200$ и с $n = 2$ признаками. В первой выборке существует единственная наилучшая закономерность и эффект расслоения проявляется максимально четко. Вторая выборка несколько зашумлена, в ней также существует

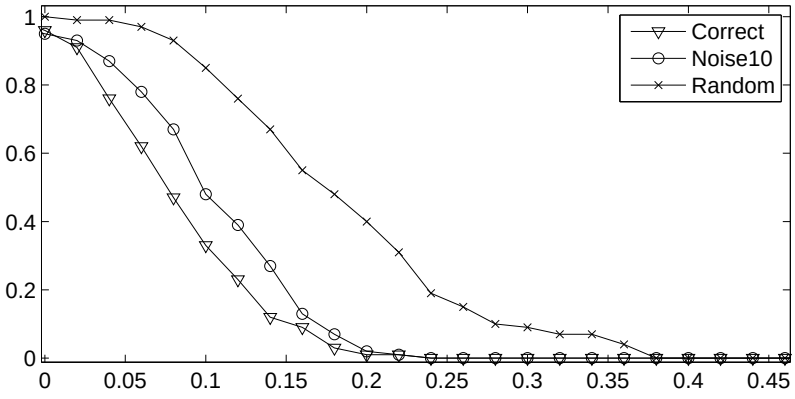

Рис. 2.6: Зависимость $\hat{Q}_\varepsilon$ от ε для трёх выборок.

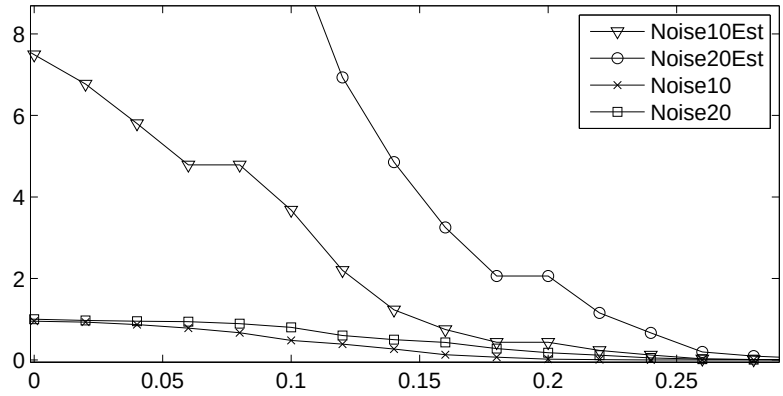

Рис. 2.7: Сравнение зависимостей $\hat{Q}_\varepsilon$ и Q_ε от ε для выборок «Noise10» и «Noise20».

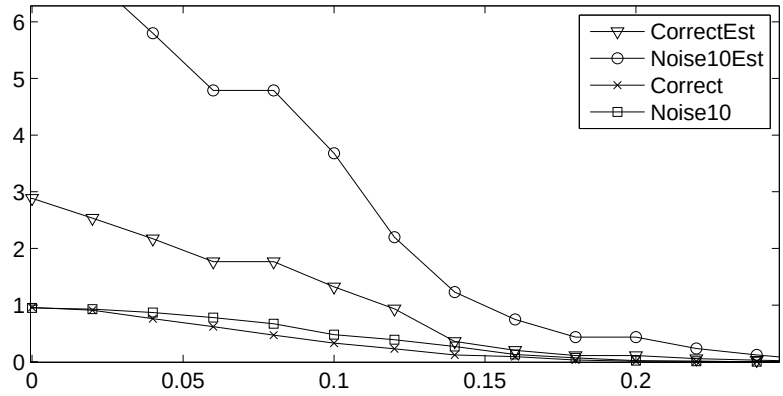

Рис. 2.8: Сравнение зависимостей $\hat{Q}_\varepsilon$ и Q_ε от ε для выборок «Correct» и «Noise10».

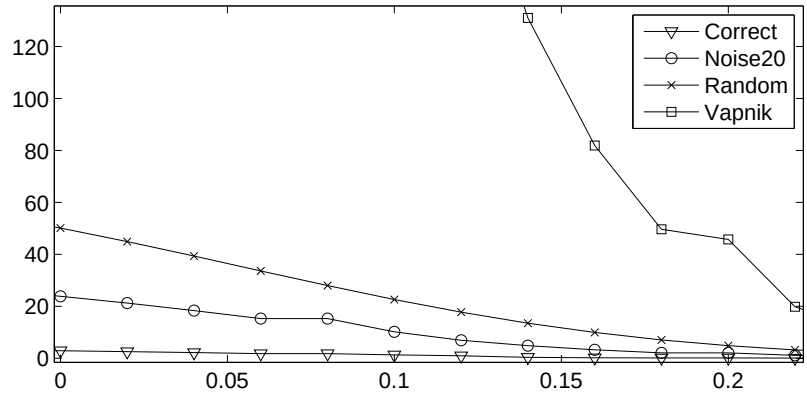

Рис. 2.9: Зависимость верхней оценки Q_ε, вычисленной по формуле (2.4), от ε, для выборок «Correct», «Noise20» и «Random».

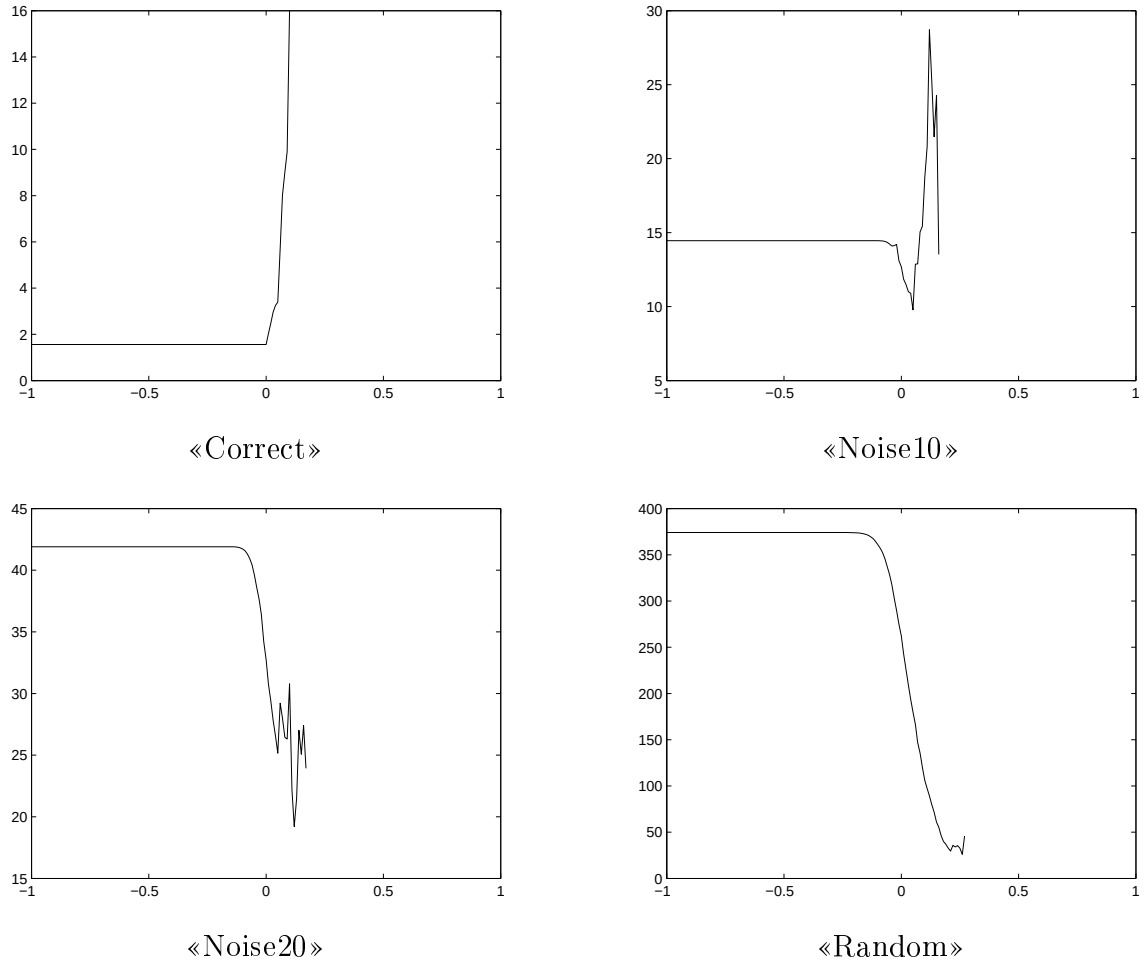

Рис. 2.10: Зависимость отношения $\hat{Q}$ к Q от ε для выборок «Correct», «Noise10», «Noise20», «Random».

наилучшая закономерность, но эффект расслоения выражен значительно слабее.

Модельные выборки подобраны таким образом, чтобы при обучении по случайным подвыборкам длины $\ell = 100$ информативности найденных в обеих выборках закономерностей в среднем совпадали. На контрольных подвыборках длины $k = L - \ell$, информативность найденной закономерности в первой подвыборке оказывается заметно выше, чем во второй. Результат представлен в таблице 2.2. Таким образом, метод оптимизации информативности сильнее переобучается

Выборка критерий инф.	1 обычный	2 обычный	1 модиф.	2 модиф.
Инф. на обучении	40,29	40,94	25,05	22,85
Инф. на контроле	39,13	34,46	39,13	34,06

Таблица 2.2: Результаты применения модифицированного критерия информативности.

на второй выборке. Однако стандартные критерии информативности не позволяют выявить эти различия. После введения поправки описанным выше методом модифицированный критерий информативности начинает надёжно отличать более информативную выборку от менее информативной.

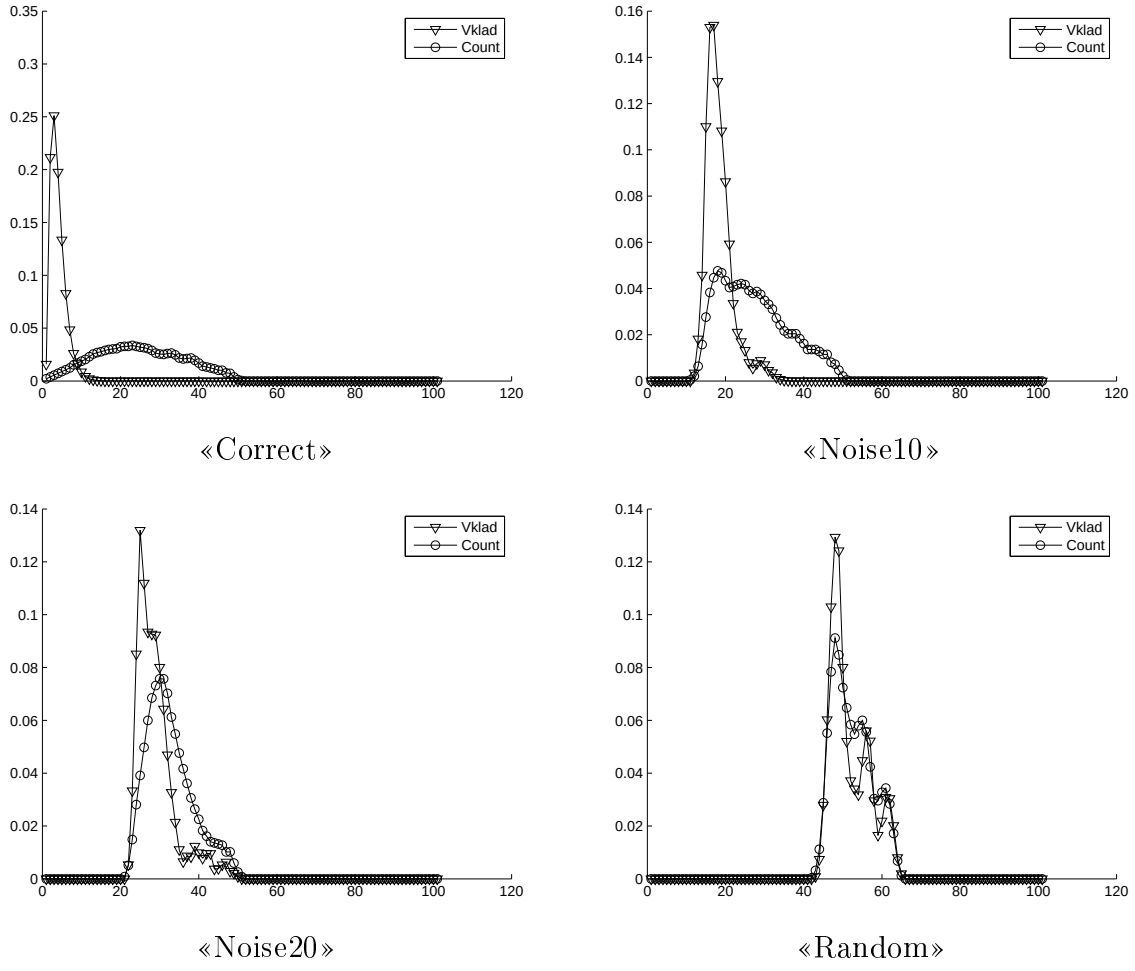

Рис. 2.11: Зависимость относительного вклада в функционал $\hat{Q}$ и доли правил от числа ошибок на полной выборке, для выборок «Correct», «Noise10», «Noise20», «Random».

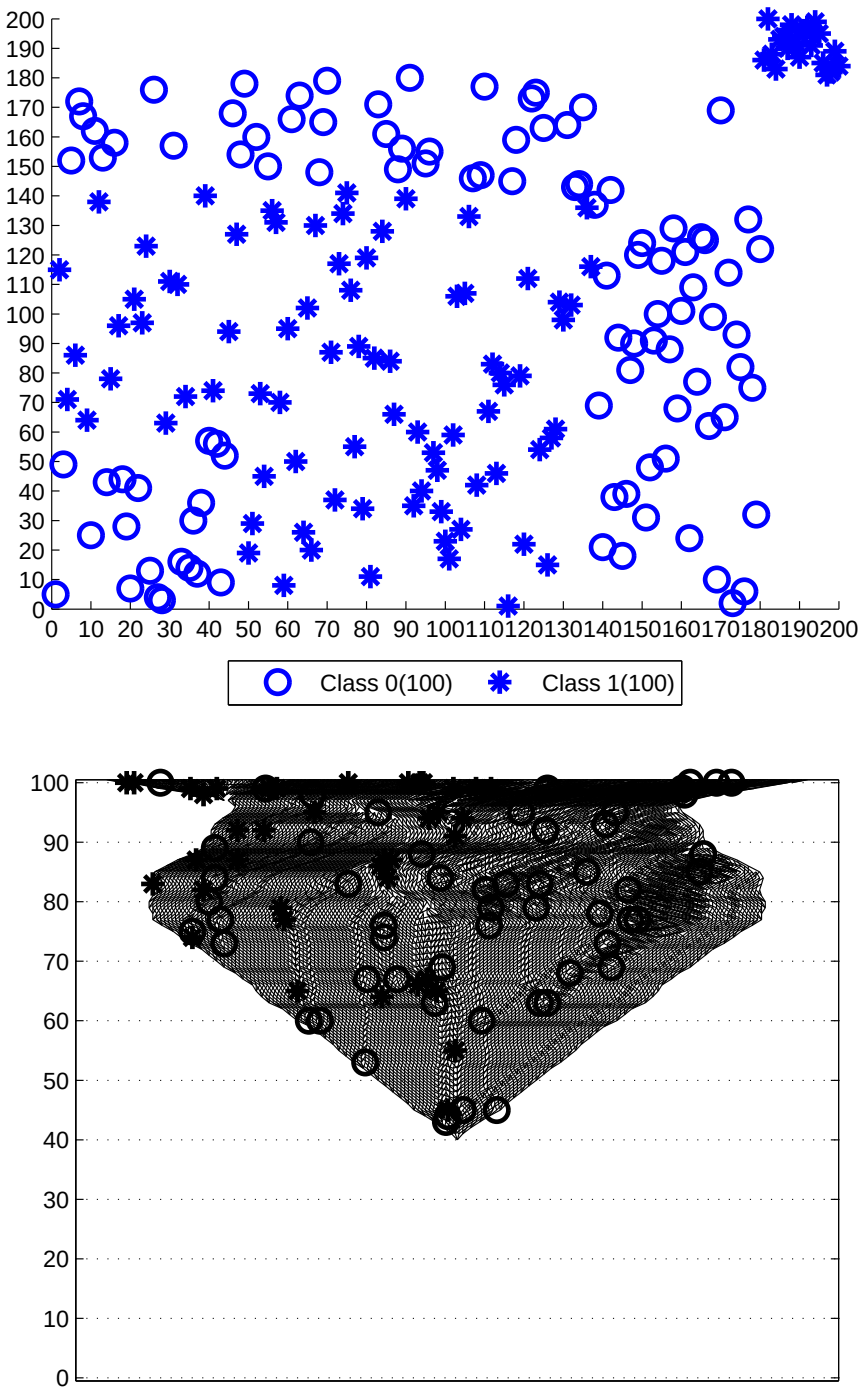

Рис. 2.12: Выборка 1 и её граф расслоения–связности.

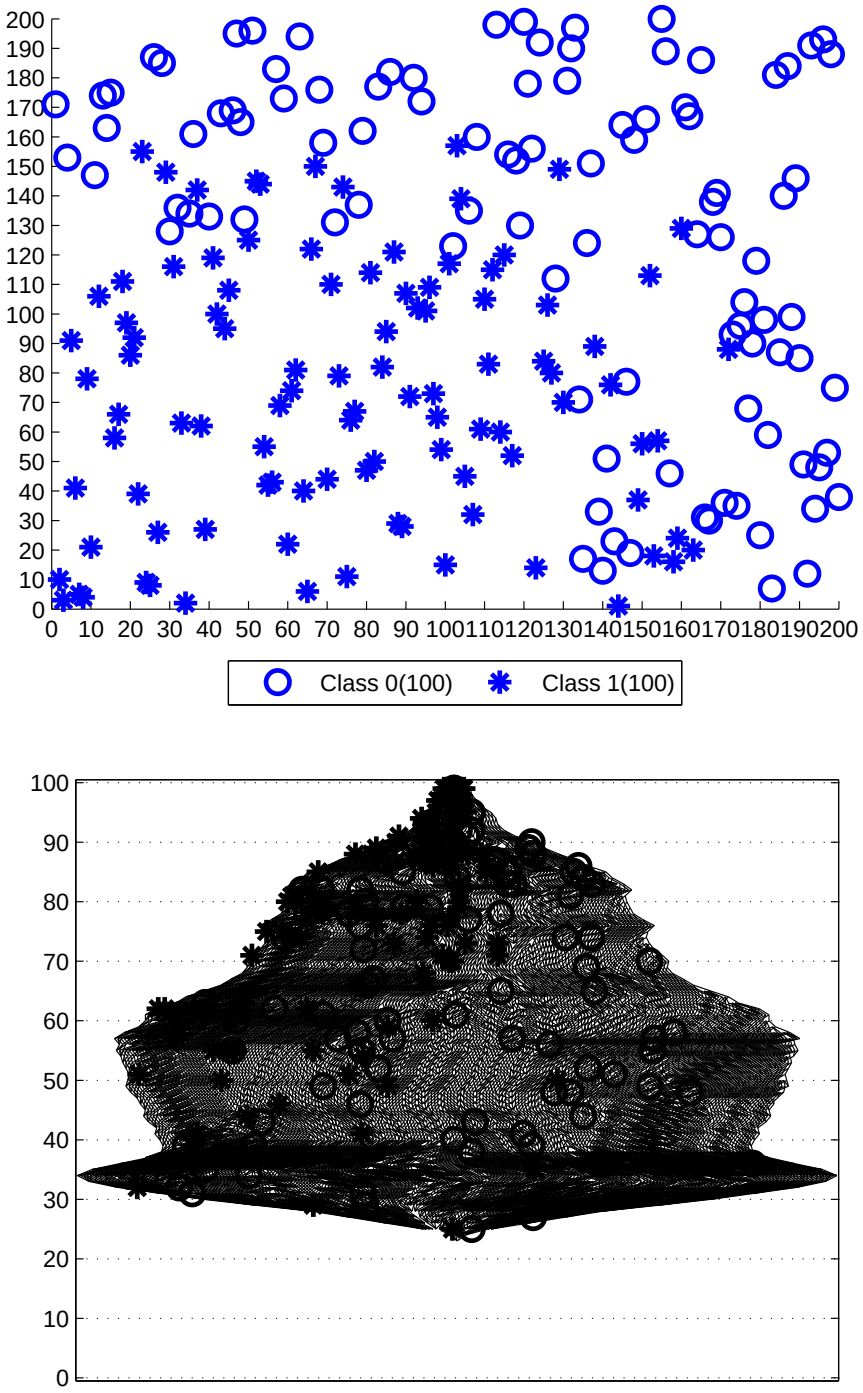

Рис. 2.13: Выборка 2 и её граф расслоения–связности.

Глава 3. Методы построения логических алгоритмов классификации

3.1 Методы оценивания информативности правил

В этом разделе описываются различные критерии информативности логических закономерностей. Основное свойство логической закономерности было описано в разделе 1.3. Будем рассматривать множество правил R_y, выделяющих объекты из некоторого фиксированного класса y. Для этого раздела введем следующие сокращенные обозначения:

P — число объектов класса y в выборке X;

p — из них число объектов, для которых $r(x) = 1$;

N — число объектов всех остальных классов $Y \backslash y$ в выборке X;

n — из них число объектов, для которых $r(x) = 1$.

Логической закономерностью является правило с достаточно большим p и достаточно малым n.

3.1.1 Эвристический пороговый ε, δ-критерий

Будем называть правило r закономерностью класса y, если он выделяет достаточно мало «чужих» объектов:

$$\frac{n}{p+n} \leqslant \varepsilon,$$

а «своих» объектов покрывает достаточно много:

$$\frac{p}{|X|} \geqslant \delta.$$

Если $n = 0$, то закономерность называется *непротиворечивой*.

3.1.2 Гипергеометрический критерий (точный тест Фишера)

Основываясь на технике проверки статистических гипотез можно ввести скалярную характеристику информативности. Пусть $\mathbb{X}$ — вероятностное пространство, выборка $X = \{x_i\}_{i=1}^{L}$ случайная и независимая, y_i и $r(x_i)$ — случайные величины. Если верна гипотеза о статистической независимости событий «$y_i = y$» и «$r(x_i) = 1$», то вероятность реализации пары (p, n) подчиняется гипергеометрическому распределению [37]:

$$h_{P,N}(p,n) = \frac{C_P^p C_N^n}{C_{P+N}^{p+n}}, \quad 0 \leqslant p \leqslant P, \quad 0 \leqslant n \leqslant N, \tag{3.1}$$

где $C_m^k = \frac{m!}{k!(m-k)!}$ — биноминальные коэффициенты, $0 \leqslant k \leqslant m$.

Если вероятность (3.1) мала, и тем не менее пара (p, n) реализовалась, то гипотеза о независимости должна быть отвергнута. Чем меньше значение вероятности, тем более значимой является связь между y_i и $r(x_i)$.

Информативностью правила r относительно класса y на выборке X определим как $H_y(r, X) = -\ln h_{P,N}(p, n)$.

Правило $r(x)$ будем называть *статистической* закономерностью для класса y, если $H_y(r, X) \geqslant H_0$ при заданном достаточно большом H_0.

Порог информативности H_0 выбирается так, чтобы ему соответствовало достаточно малое значение вероятности. Описанный критерий применяется в статистике для проверки гипотезы о независимости событий и называется *точным тестом Фишера* (Fisher's exact test, FET).

Замечание 3.1. Значение информативности данного критерия существенно зависит от пары (P, N). Это может быть неудобно, например при рассмотрении выборки с большим число пропусков в данных. В этом случае на ряде объектов правило не может быть вычислено и, логично, исключить эти объекты при оценке информативности правила вовсе. Можно нормировать значения p, n, P, N на некоторую наперед заданную константу.

Замечание 3.2. Вычислять $H_y(r, X)$ можно весьма эффективно, если заранее подготовить таблицу логарифмов факториалов $L_k = \ln k!$. Эта таблица вычисляется по рекурентной формуле $L_1 = 0;\ L_k = L_{k-1} + \ln k$. Информативность $H_y(r, X)$ представляется как сумма 9 элементов данной таблицы.

3.1.3 Энтропийный критерий и индекс Джини

Асимпотитческим приближением гипергеометрического критерия является *энтропийный* критерий информативности IGain [49]:

$$IGain_y(r, X) = h\left(\frac{P}{P+N}\right) - \frac{p+n}{|X|}h\left(\frac{p}{p+n}\right) - \frac{|X|-p-n}{|X|}h\left(\frac{P-p}{|X|-p-n}\right),$$

где $h(q) = -q\log_2 q - (1-q)\log_2(1-q)$ — функция энтропии пары исходов с вероятностями q и $1-q$.

На практике используют также индекс Джини (Gini impurity), который отличается только функцией $h(q) = 2q(1-q)$, которая очень близка к функции энтропии [34].

3.1.4 Парето-оптимальный фронт по паре критериев (p, n)

Одним из способов многокритериальной оптимизации является нахождение Парето-оптимального фронта. Правило r, которое покрывает p_r «своих» объектов и n_r «чужих» принадлежит к оптимальному фронту, если не существует правила q, такого что $p_r \leqslant p_q$ и $n_r \geqslant n_q$, см. рис. 3.1.

При рассмотрении на примерах оказывается, что удобнее использовать выпуклую оболочку набора правил. Выпуклые оболочки строятся последовательно, построив первую удаляем правила из списка, строим вторую и так далее. Последовательность выпуклых оболочек строится с помощью алгоритма обхода Грэхэма, см. рис. 3.2.

Данный метод отбора можно комбинировать со многими другими алгоритмами.

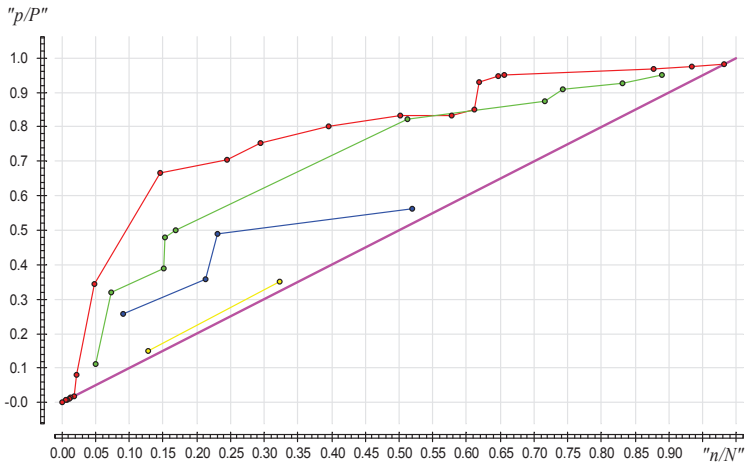

Рис. 3.1: Парето-слои набора правил. Разными цветами обозначены разные слои.

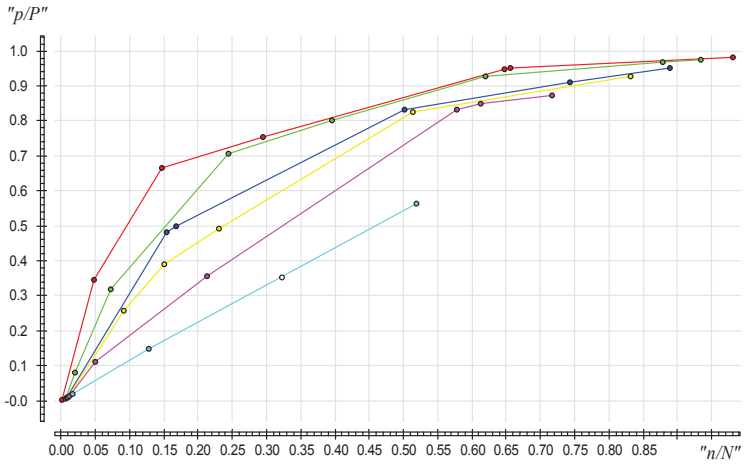

Рис. 3.2: Выпуклые оболочки набора правил. Разными цветами обозначены разные выпуклые оболочки.

3.2 Методы поиска информативных конъюнкций

В данном разделе речь пойдет о поиске правил в виде описанном в разделе1.3. Для различных типов признаков используются следующие виды термов:

- для номинальных, бинарных и порядковых признаков:
  - $x^j = \theta$,
  - $x^j \neq \theta$
- для количественных:
  - $x^j \geqslant \theta$,
  - $x^j \leqslant \theta$,
  - $\theta \leqslant x^j \leqslant \theta'$,
  - $x^j \in (-\infty; \theta] \cup [\theta'; \infty)$
- для всех:
  - Значение x^j заданно,
  - Значение x^j не заданно

где θ и θ' — некоторые константы, называемые *порогами*. Допустимые значения порогов для j-ого признака будем обозначать $\Theta_j = \{\theta_{jk}\}$. Обозначим через ξ_{jk} интервал значений j-ого признака $(\theta_{jk}; \theta_{jk+1})$.

3.2.1 Бинаризация исходной информации

Определение 3.1. *Градацией порядкового или количественного признака f_j будем называть функцию $C_j : \Theta_j \mapsto \widetilde{\Theta}_j$, которая уменьшает количество порогов, $|\widetilde{\Theta}_j| < |\Theta_j|$.*

Градуирование используется для уменьшения числа допустимых термов, а следовательно, сокращения перебора и увеличения быстродействия алгоритмов поиска закономерностей.

Распространим понятие информативности на градацию, будем рассматривать интервалы градации как простые правила, покрывающее объекты попадающие в эти интервалы.

Описание алгоритма 3.2.1. Сначала выделим «чистые» интервалы, т.е. интервалы, содержащие объекты только одного класса, затем начнем объединять соседние интервалы, удаляя из множества $\widetilde{\Theta}$ по одному порогу. Выбор удаляемого порога осуществляется жадным способом. Выбирается тот порог, при удалении которого будет получено максимальное увеличении (минимальное уменьшение) информативности нового интервала относительно старых. Удаление производится до тех пор, пока не выполнится критерий останова.

Критерий останова алгоритма градации. Можно использовать различные критерии для останова алгоритма градации. Например, достижение заданного количества интервалов N или неотрицательность выигрыша при слиянии очередных двух интервалов: $H(\xi_{i-1} \cup \xi_i) - \max\{H(\xi_{i-1}), H(\xi_i)\} \leqslant 0$.

Замечание 3.3. Локальная оптимизация порогов $\widetilde{\Theta}$ заключается в следующем. Для каждого порога нужно выбрать такое положение между соседними в котором сумма информативностей двух интервалов, границей для которых служит оптимизируемый порог, максимальна.

Замечание 3.4. Представленный алгоритм имеет сложность порядка $O(\ell^2)$. При построении градации признака на N интервалов по большой выборке этот алгоритм оказывается не эффективным. Вместо удаления по одному порогу нужно, сразу после объединения объектов в одноклассовые блоки, разбить числовую ось на N интервалов примерно одинаковых по количеству попавших в них объектов. Затем локально оптимизируется каждый порог. Сложность такого алгоритма составляет $O(N\ell)$.

Алгоритм 3.2.1 Алгоритм градуирования порядкового или количественного признака

Вход:

X^ℓ — обучающая выборка;

j — индекс градуируемого признака.

Выход:

$\widetilde{\Theta}$ — сокращенное множество допустимых значений порогов.

1: Начальное приближение, $\widetilde{\Theta} = \{\frac{x_i^j + x_{i+1}^j}{2}\}_{i=1}^{\ell-1}$;

2: Слияние соседних зон, содержащих объекты одного и того же класса в блоки, $\widetilde{\Theta} = \{\theta_k | \theta_{k-1} < x_i^j < \theta_k \Rightarrow y_i = c_k\}$;

3: **повторять**

4: **для всех** интервалов ξ_k

5: Вычислить выигрыш от слияния интервалов, $\delta H_k = H(\xi_{k-1} \cup \xi_k) - \max\{H(\xi_{k-1}), H(\xi_k)\}$;

6: Выбрать порог, удаление которого даст наибольший выигрыш по информативности, $k_0 = \arg\max_k \delta H_k$;

7: Удалить этот порог, $\widetilde{\Theta} = \widetilde{\Theta} \backslash \theta_{k_0}$;

8: **пока** не выполнится критерий останова

9: Локальная оптимизация порогов;

10: **вернуть** $\widetilde{\Theta}$;

Замечание 3.5. Увеличить интепретируемость порогов можно с помощью задания пользователем таблицы точности на числовой оси. Например, для признака «возраст» может иметь смысл использовать только целые числа, или даже различную точность в разных интервалах (в интервале от 20 до 50 лет рассматривать пороги с точностью 1 год, а в интервале от 50 лет и выше — 5 лет). Также пользователь может задать пороги которые должны быть включены обязательно (например, 0).

3.2.2 Стабилизация набора правил

В процессе построения правил обычно довольно трудоёмко (как вычислительно, так и алгоритмически) следить за уникальностью получаемых правил. Ещё сложнее выявлять похожие правила (например, правила различающиеся порогом в одном из термов). Решением этих проблем является *стабилизация* набора построенных правил. Процесс стабилизации позволяет получать локально не улучшаемые правила (правила информативность которых невозможно увеличить изменив любой из порогов входящих в него термов). Похожие правила преобразуются в одно, а дубликаты можно легко удалить.

Описание алгоритма 3.2.2. Сначала отсортируем правила по информативности, это позволит эффективно пропустить правила, которые уже проходили стабилизацию (см. замечание 3.6, здесь используется равенство информативностей у одинаковых правил). Затем попытаемся заменить каждый терм в правиле на другой с таким же признаком. Этим действием мы выбираем оптимальные пороги для каждого терма. Возможно, что терм данного признака и вовсе лучше удалить из правила. Более того, пользователь может назначить поощрение за более короткие правила, ведь их проще интерпретировать и они более общи. Если информативность правила уменьшится не более чем в α раз при удалении терма, то удаляем терм.

Стабилизированное правило добавляем в набор, если такового там еще нет и если правило удовлетворяет некоторым условиям которые проверяет функция $Valid(r)$.

Функция $Valid(r)$**.** Пользователь алгоритма стабилизации может иметь какие-то заранее известные требования к искомым правилам. Функция $Valid(r)$ проверяет правило на соответствие этим требованием. В работе используются следующие требования к правилам:

- допустимый класс правила (запрещается генерация правил конкретных

классов);

- минимальное число покрываемых правилом объектов (для каждого класса задается отдельно);
- минимальная информативность правила (для каждого класса задается отдельно);
- максимально допустимая доля ошибок правила (для каждого класса задается отдельно).

Замечание 3.6. Два правила считаются равными, когда они покрывают одни и те же объекты на обучающей выборке. Сравнивать множества покрытия обоих правил может оказаться трудоёмко. Чтобы избежать этого можно сначала проверить необходимые условия равенства правил (равенство: информативностей, количества покрываемых объектов своего/чужого классов, количества термов).

Замечание 3.7. Можно увеличить эффективность алгоритма 3.2.2, если для числовых и порядковых признаков просматривать не все термы при выборе лучшего. Можно поочередно двигать левый и правый пороги (константы θ и θ'). В этом случае сложность перебора с квадратичной снизится до линейной по количеству допустимых порогов.

3.2.3 Случайный поиск

Описание алгоритма 3.2.3. Сгенерируем K случайных правил. Под случайным правилом будем понимать правило случайной, но не превышающей M, длины, с термами случайно выбранных признаков. Затем проведем стабилизацию (алгоритм 3.2.2) полученных правил, и выберем H лучших (см. раздел 3.1).

Функция $Random(n)$ выдает случайное целое число из диапазона $\overline{1,n}$. Вероятности получения любого из чисел одинакова и равна $\frac{1}{n}$.

Достоинства алгоритма: Простота реализации, скорость работы, нахождение существенно различных правил.

Недостатки алгоритма: Нечувствительность к весам объектов.

3.2.4 Случайный поиск с адаптацией

Описание алгоритма 3.2.4. Представленный алгоритм это многократное повторение алгоритма 3.2.3 с поощрением информативных признаков и термов. Признак или терм считаются «хорошими», если они встречаются в хороших правилах, и «плохими» если в плохих. Увеличивая вероятность появления в следующей итерации «хороших» признаков или термов, мы тем самым пытаемся улучшить получаемый в итерации набор правил.

Функция $RandomP(n, P)$ это функция выдающая число из диапазона $\overline{1,n}$ с вероятностью P_i для i-ого числа. Должны выполняться следующие условия: $n = |P|$; $\sum_{i=1}^{n} P_i = 1$; $P_i > 0$, для всех $i \in \overline{1,n}$.

Функция $[P, Q] = RecalcP(R)$ пересчитывает вектор вероятности выбора признака P и матрицу вероятностей выбора каждого из допустимых термов каждого признака Q, в зависимости от полученного на текущем этапе набора правил R. Общая идея этой функции — поощрить «хорошие», встретившиеся в «хороших» правилах, признаки и термы, и оштрафовать «плохие». Для этого отсортируем по одному из критериев качества (см. раздел 3.1) все правила из множества R. Возьмем по K_0 правил из начала («хорошие») и конца («плохие») списка. Умножим на α_0 вероятности P_i для тех признаков f_i, что попали в K_0 «хороших» правил и разделим на α_0 вероятности тех признаков что попали в K_0 «плохих» правил. Отнормируем полученный вектор. Аналогично поступим и для матрицы вероятностей.

Замечание 3.8. Полученный в конце работы алгоритма 3.2.4 вектор P можно

использовать для ранжирования признаков по полезности.

Достоинства алгоритма: Простота реализации, скорость работы, нахождение различных правил.

Недостатки алгоритма: Нечувствительность к весам объектов.

3.2.5 Усеченный поиск в ширину

Графы часто используются для решения оптимизационных задач. Прежде всего графы позволяют наглядно представить имеющуюся информацию, кроме того, для графов существует обширная теория со множеством хорошо изученых алгоритмов отыскания объектов с оптимальными характеристиками.

Представим все множество правил в виде дерева. Узлы дерева — правила, связи есть только между теми правилами которые можно получить одно из другого добавлением или удалением одного терма. Корнем такого дерева (нулевым уровнем) является пустое правило (правило покрывающее все объекты в выборке). На первом уровне находятся однотермовые правила, на втором двухтермовие и т.д. Связи есть только между соседними уровнями, внутри уровней связей нет.

Описание алгоритма 3.2.5. Начинаем с корня дерева (пустого правила), считая его текущим набором правил. На каждом шаге алгоритма добавляем к правилам из текущего набора (их не более чем S_1) по одному терму всевозможными способами (выбираем из следующего уровня дерева). Выбираем не более чем S_2 правил-потомков которые лучше правила-предка. Если правило-предок не породило более хороших потомков, то это правило добавляется в конечный список. Из получившихся правил-потомков выбирается S_1 новых правил-предков.

Достоинства алгоритма: Возможность использовать взвешенные объекты.

Недостатки алгоритма: Нахождение похожих правил.

Алгоритм 3.2.2 Алгоритм стабилизации набора правил

Вход:

X^ℓ — обучающая выборка;

R — множество правил для стабилизации;

T_j — набор допустимых термов для j-ого признака ($j = \overline{1,n}$);

α — параметр поощряющий короткие правила, $\alpha \in (0;1]$.

Выход:

R_{stable} — стабилизированное множество правил.

Вычислить информативность всех правил из R на выборке X^ℓ;
Отсортировать множество R по информативности;
Инициализировать множество $R_{stable} = \varnothing$;
для всех правил $r_k \in R$, где r_k задано множеством термов $r_k = \{t_1 \ldots, t_{M_k}\}$
  если r_k совпадает с r_{k-1} **то**
    следующий ;
  повторять
    для всех $t \in r_k$
      $r_k = r_k \backslash t$;
      $t' = \arg\max\limits_{t \in T_i} H(r_k \cup t)$;
      если $H(r_k) \cdot \alpha < H(r_k \cup t')$ **то**
        $r_k = r_k \cup t'$;
  пока правило r_k меняется
  если $r_k \notin R_{stable}$ и $Valid(r_k)$ **то**
    Добавить правило r_k в R_{stable};
вернуть R_{stable};

Алгоритм 3.2.3 Алгоритм случайного поиска

Вход:

$F = \{f_i\}_{i=1}^{n}$ — набор признаков;

T_j — набор допустимых термов для j-ого признака ($i = \overline{1,n}$);

K — количество генерируемых правил;

H — количество отбираемых правил;

M — максимальная длина правила.

Выход:

$R = \{r_k\}_{k=1}^{H}$ — набор правил.

Инициализировать набор правил, $R = \varnothing$;
для всех $k = 1, \ldots, K$
    Выбирать случайным образом длину правила, $M_k = Random(M)$;
    Создать пустое правило, $r_k = \varnothing$;
    для всех $l = 1, \ldots, M_k$
        Выбирать признак, термов которого еще нет в правиле r_k, $i = Random(n)$;
        Выбирать терм из множества допустимых термов $j = Random(|T_i|)$;
        Добавить к текущему правилу j-ый терм из T_i, $r_k = r_k \cup t$;
    Добавить правило в набор, $R = R \cup r_k$;
$R = Stabilize(R)$;
$R = GetBestRules(R, H)$;
вернуть R;

Алгоритм 3.2.4 Алгоритм случайного поиска с адаптацией

Вход:

$F = \{f_i\}_{i=1}^{n}$ — набор признаков;

T_i — набор допустимых термов для i-ого признака ($i = \overline{1,n}$);

M — максимальная длина правила;

K — количество генерируемых правил;

H — количество отбираемых правил;

L — количество итераций.

Выход:

$R = \{r_k\}_{k=1}^{H}$ — набор правил.

1: Инициализировать вероятности выбора признаков, $P_i = \frac{1}{n}$, $i = \overline{1,n}$;
2: Инициализировать вероятности выбора термов из множества допустимых для каждого признака, $Q_{ij} = \frac{1}{|T_i|}$, $i = \overline{1,n}$, $j = \overline{1,|T_i|}$;
3: Инициализировать список правил, $R = \varnothing$;
4: **для всех** $l = 1, \ldots, L$
5: Инициализировать набор правил, $R_l = \varnothing$;
6: **для всех** $k = 1, \ldots, K$
7: Выбрать длину правила $M_k = Random(M)$;
8: Инициализировать правило, $r_k = \varnothing$;
9: **для всех** $m = 1, \ldots, M_k$
10: Выбрать номер добавляемого признака, $i = RandomP(n, P)$;
11: Выбрать номер терма во множестве допустимых термов T_i для i-ого признака, $j = RandomP(|T_i|, Q_i)$;
12: $r_k = r_k \bigvee t_{ij}$;
13: Добавить к набору полученное правило, $R_l = R_l \cup r_k$;
14: $R_l = Stabilize(R_l)$;
15: $[P, Q] = RecalcP(R_l)$;
16: $R = R \cup R_l$;
17: $R = Stabilize(R)$;
18: $R = GetBestRules(R, H)$;
19: **вернуть** R;

Алгоритм 3.2.5 Алгоритм усеченного поиска в ширину

Вход:

T_i — набор допустимых термов для i-ого признака ($i = \overline{1,n}$);

M — максимальная длина правила;

S_1 — количество правил оставляемых на каждом этапе для улучшения;

S_2 — количество улучшенных от каждого из S_1 правил;

H — количество отбираемых правил;

Выход:

$R = \{r_k\}_{k=1}^{H}$ — набор правил.

Инициализировать набор правил, $R = \varnothing$;
Инициализировать набор правил пустым правилом, $R_0 = \{\varnothing\}$;
для $m = 1, \ldots, M$
    Инициализировать набор правил m-ого шага, $R_m = \varnothing$;
    для всех $r_k \in R_{m-1}$
        Инициализировать набор правил получающихся добавлением одного терма к правилу r_k, $Q_k = \varnothing$;
        для всех признаков f_i, не входящих в термы r_k
            для всех $t \in T_i$
                Добавить в Q_k правило $r_k \cup t$;
        Оставить в Q_k правила с информативностью большей чем у r_k, $Q_k = \{r | r \in Q_k, H(r) > H(r_k)\}$;
        Выбрать S_2 лучших, $Q_k = GetBestRules(Q_k, S_2)$;
        если $Q_k = \varnothing$ **то**
            Сохранить r_k как неулучшаемое в конечном списке правил, $R = R \cup r_k$;
        иначе
            Сохранить улучшенные правила в наборе правил этого шага, $R_m = R_m \cup Q_k$;
    если $m \neq M$ **то**
        $R_m = GetBestRules(R_m, S_1)$;
$R = Stabilize(R_M)$;
$R = GetBestRules(R, H)$;
вернуть R;

3.3 Методы построения композиций информативных правил

3.3.1 Решающий список

Решающий список (decision list, DL) — это наиболее простой логический алгоритм, как по своей структуре, так и по способу построения.

Определение 3.2. *Решающий список задаётся последовательностью правил — предикатов $r_1(x), \ldots, r_p(x)$, с соответсвующими им ответами $c_1, \ldots, c_p, c_0$ и функционирует следующим образом:*

Итак, получив на входе объект x, алгоритм проверяет правила последовательно, и как только находится правило $r_i(x) = 1$, возвращает ответ c_i. Если ни одно из правил не смогло классифицировать объект, возвращается ответ c_0, означающий отказ алгоритма от классификации объекта x. На практике такие объекты часто приписывают классу, имеющему минимальную цену ошибки. Например, в задаче выдачи кредитов отказ алгоритма приведёт, скорее всего, к более осторожному решению «не выдавать».

Соседние правила в списке r_{i-1}, r_i можно переставлять местами, если они помечены одним классом, $c_{i-1} = c_i$. В общем случае перестановка правил изменяет алгоритм.

Для лучшей интерпретруемости можно строить правила сначала одного класса, потом другого и т.д. Рассмотрим первое и второе правила в решающем списке. Пусть первое правило должно выделять объекты класса c_1, тогда это правило будет «хорошим», если оно покроет много объектов класса c_1 и мало объектов других классов. Второе правило из решающего списка выделяет объекты класса c_2. Аналогично, оно будет «хорошим», если покрывает много объектов класса c_2 и мало объектов других классов, однако ничто не мешает ему покрывать и объекты, уже покрытые первым правилом. Получается что если $c_1 \neq c_2$, то второе правило может не является «хорошей» логической закономерностью само по себе. Если

Алгоритм 3.3.1 Алгоритм классификации по решающему списку

1: **для всех** $i = 1, \ldots, p$
2: **если** $r_i(x)$ **то**
3: **вернуть** c_i;
4: **вернуть** c_0;

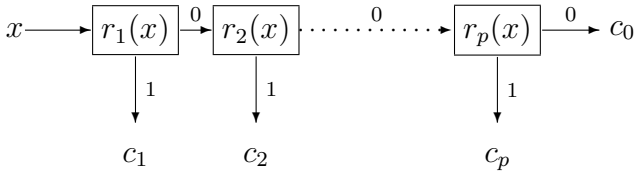

Рис. 3.3: Решающий список.

же мы потребуем от алгоритма синтеза решающего списка построить сначала все правила, выделяющие один класс, затем правила, выделяющие другие классы, то получим решающий список, состоящий из правил, которые могут быть легко интерпретированы экспертами.

3.3.2 Взвешенное голосование

Определение 3.3. *Взвешенное голосование правил задаётся списком правил — предикатов* $r_1(x), \ldots, r_p(x)$, *с соответсвующими им ответами* $c_1, \ldots, c_p \in Y$ *и весами* $w_1, \ldots, w_p \in \mathbb{R}$. *Решение о принадлежности объекта к какому-либо классу принимается так:* $a(x) = \arg\max_{y \in Y} \sum_{i=1}^{p} w_i r_i(x)[c_i = y]$

Такой способ взаимодействия правил позволяет компенсировать ошибки одних правил верными ответами других, поэтому требования к доле ошибок правил для этого алгоритма мягче.

Бустинг(boosting) был изобретён американскими учёными Френдом и Шапиром как универсальный метод построения выпуклой комбинации классификаторов [40]. Он сразу стал очень популярен благодаря простоте, эффективности и низкой склонности к переобучению. Он позволяет практически неограниченно наращивать сложность алгоритма, объединяя сотни и тысячи классификаторов

Алгоритм 3.3.2 Алгоритм построения решающего списка

Вход:

X^ℓ — обучающая выборка;

p — желаемая длина решающего списка;

Выход:

$DL = \{(r_1, c_1), \ldots, (r_p, c_p), c_0\}$ — решающий список;

Инициализировать список правил, $DL = \varnothing$;
Инициализировать множество объектов обучения, $U = X^\ell$;
пока $|DL| < p$
    $\Phi = GetRules(U)$;
    если $(\Phi = \varnothing)$ **то**
        вернуть DL
    $r = GetBestRule(\Phi)$;
    Добавить r в DL;
    Удалить из U объекты, покрытые выбранным правилом:
    $U = \{x \in U : r(x) = 0\}$;
вернуть DL

(в нашем случае — закономерностей) по принципу взвешенного голосования.

Основная идея бустинга очень проста. Веса вводятся не только для алгоритмов, но и для объектов. Закономерности строятся последовательно, и после построения очередной закономерности веса покрытых ею объектов изменяются — уменьшаются у объектов своего класса и увеличиваются у объектов всех остальных классов. Обновлённый вектор весов w нормируется и используется при поиске следующей закономерности φ по критерию максимума взвешенной информативности. В результате каждая следующая закономерность сосредоточивается на покрытии объектов, «наименее покрытых» предыдущими закономерностями. Это способствует усилению различий между закономерностями и наиболее равномерному покрытию объектов.

Описанная стратегия напоминает алгоритм построения решающего списка. Разница в том, что там было достаточно покрыть объект один раз, после чего он исключался из рассмотрения. Здесь же только уменьшается его вес.

Для реализации этой идеи остаётся понять, как именно должны вычисляться веса объектов и веса закономерностей на каждом шаге алгоритма.

Упростим немного задачу и ограничимся пока случаем двух классов. Договоримся помечать классы числами $\{-1, +1\}$ вместо обычного $\{0, 1\}$.

Экспоненциальная аппроксимация пороговой функции потерь Пусть уже построено $T = T_{-1} + T_{+1}$ закономерностей. При добавлении ещё одной закономерности $\varphi_c(x)$ в список R_c взвешенная сумма голосов за класс c примет вид

$$\Gamma'_c(x) = \Gamma_c(x) + \alpha\varphi_c(x).$$

Задача состоит в том, чтобы найти закономерность φ_c и её вес α, при которых алгоритм $a(x)$ допускает минимальное число ошибок на обучающей выборке X^ℓ.

Число ошибок перед добавлением закономерности φ_c:

$$Q_T = \sum_{i=1}^{\ell} \left[\Gamma_{-y_i}(x_i) - \Gamma_{y_i}(x_i) > 0\right].$$

Число ошибок после добавления закономерности φ_c:

$$Q_{T+1}(\varphi_c, \alpha) = \sum_{i=1}^{\ell} [y_i = c] \left[\Gamma_{-y_i}(x_i) - \Gamma_{y_i}(x_i) - \alpha\varphi_c(x_i) > 0\right] + \\ + \sum_{i=1}^{\ell} [y_i \neq c] \left[\Gamma_{-y_i}(x_i) + \alpha\varphi_c(x_i) - \Gamma_{y_i}(x_i) > 0\right].$$

Выписанный функционал содержит параметр α внутри пороговой функции вида $[z(\alpha) > 0]$ и является разрывной функцией от α. Минимизация такого функционала сводится к сложной комбинаторной задаче. Будем решать её приближенно. Заменим пороговую функцию непрерывно дифференцируемой оценкой сверху. Тогда минимизацию по α можно будет выполнить аналитически.

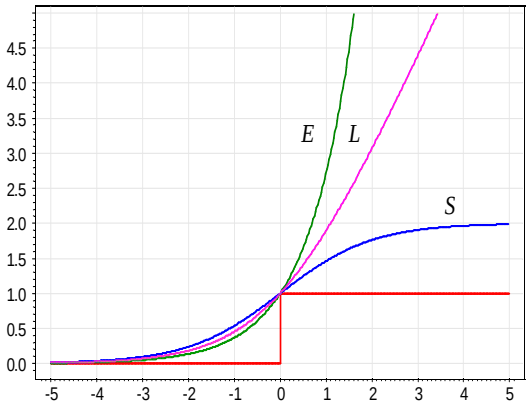

Рис. 3.4: Гладкие аппроксимации пороговой функции потерь: E — экспоненциальная, S — сигмоидная, L — логарифмическая.

Выбор конкретной аппроксимирующей функции является эвристикой. В частности, можно воспользоваться любым из трёх неравенств, справедливых при всех действительных z, см. рис. 3.4:

$$[z > 0] \leqslant \begin{cases} e^z & \text{— экспоненциальная аппроксимация;} \\ \dfrac{2e^z}{1+e^z} & \text{— сигмоидная аппроксимация;} \\ \log_2(1+e^z) & \text{— логарифмическая аппроксимация.} \end{cases}$$

Получится, соответственно, три варианта алгоритма бустинга (разумеется, их может быть гораздо больше). Остановимся на первом, как наиболее простом.

Запишем верхнюю оценку $\bar{Q}_T$ функционала Q_T:

$$Q_T \leqslant \bar{Q}_T \equiv \sum_{i=1}^{\ell} \underbrace{\exp\bigl(\Gamma_{-y_i}(x_i) - \Gamma_{y_i}(x_i)\bigr)}_{w_i} = \sum_{i=1}^{\ell} w_i.$$

Величины w_i, $i = 1, \ldots, \ell$ неплохо подходят на роль тех самых весов объектов, о которых шла речь в самом начале. Действительно, если алгоритм голосования $a(x)$ правильно классифицирует объект x_i, то $w_i < 1$. Если ошибается, то $w_i > 1$. Чем больше перевес голосов в пользу ошибочного класса, тем больше вес w_i. Больший вес получают объекты, которые реже покрывались предыдущими закономерностями, и потому оказались более «трудными» для

алгоритма $a(x)$. Если следующую закономерность оптимизировать по критерию взвешенной информативности с весами w_i, то она сосредоточится на покрытии именно этих объектов. Теорема 3.1 показывает, что выбор w_i в качестве весов объектов — не просто эвристика, а является оптимальным.

Введём вектор $w = (\widetilde{w}_i)_{i=1}^{\ell}$ с компонентами $\widetilde{w}_i = w_i \ell / \bar{Q}_T$. Тогда будет выполнено условие нормировки $\frac{1}{\ell}\sum_{i=1}^{\ell} \widetilde{w}_i = 1$. [52, 36]

Теорема 3.1. *Минимум функционала $\bar{Q}_{T+1}(\varphi_c, \alpha)$ достигается при*

$$\varphi_c^* = \arg\max_{\varphi} J_c^w(\varphi, X^\ell), \qquad J_c^w(\varphi, X^\ell) = \sqrt{g_c^w(\varphi)} - \sqrt{b_c^w(\varphi)}; \tag{3.2}$$

$$\alpha^* = \frac{1}{2}\ln\frac{g_c^w(\varphi_c^*)}{b_c^w(\varphi_c^*)}, \quad \textit{если только } b_c^w(\varphi_c) \neq 0. \tag{3.3}$$

Замечание 3.9. Теорема не позволяет определить вес непротиворечивой закономерности φ_c^*, поскольку тогда знаменатель (3.3) обращается в нуль. На практике вводят дополнительный параметр $\lambda \in (0, 1)$, слегка модифицируя формулу расчёта веса:

$$\alpha^* = \frac{1}{2}\ln\frac{g_c^w(\varphi_c^*)}{\max\{b_c^w(\varphi_c^*), \lambda\}}.$$

Чем меньше λ, тем больший вес получают непротиворечивые закономерности. Например, можно положить $\lambda = \frac{1}{2}$ или $\frac{1}{4}$.

Эвристики

1. Если не найдено ни одного правила, то пересчитать веса объектов по взвешенной информативности и попробовать еще раз.

2. Целочисленные веса для ускорения счета.

3. Отбирать на каждом шаге по одному правилу каждого класса. Скорее всего эти правила имеют малое пересечение по покрываемым объектам.

4. Отбирать на каждом шаге правила из лучшего Парето-слоя, или выпуклой оболочки, см. раздел 3.1.4.

Рекомендации по настройке алгоритма 3.3.3 Остановимся подробнее на параметрах алгоритма синтеза правил для взвешенного голосования.

- α. Целочисленные операции выполняются гораздо быстрее операций с плавающей точкой. Для увеличения скорости работы алгоритма была введена целочисленная система весов объектов. Такой подход имеет и свои минусы: сокращение точности представления весов объектов. Так в некоторый момент времени вес объекта может обнулиться и объект выпадет из дальнейшего рассмотрения. Что бы избежать этого был введен минимальный вес объекта. Рекомендуемое значение $\alpha = \frac{1}{4} \div \frac{1}{2}$.

- p. Теоретически, количество ошибок на обучении можно уменьшить до 0, т.е. построенный список правил не будет ошибаться при взвешенном голосовании на обучающей выборке. Однако такой путь ведет к переобучению, т.е. к чрезмерной настройке на шумы обучающей выборке. Настроенный таким образом алгоритм будет часто ошибаться на тестовой выборке, что не очень хорошо. Кроме того, большое количество правил, обычно, трудноинтерпретируемо. Поэтому рекомендуется ограничивать число правил в списке. Рекомендуемое значение $p = 20 \div 50$.

Основной проблемой генерации набора правил для алгоритмов взвешенного голосования является синтез существенно различных конъюнкций. Здесь эта проблема решается с помощью взвешенности объектов обучающей выборки.

3.3.3 Логистическая регрессия

Постановка задачи восстановления логистической регрессии. В классической постановке [43] задача классификации с помощью логистической регрессии ставиться следующим образом. Пусть задана обучающая выборка как совокупность L пар (x_i, y_i), где $x_i \in \mathbb{R}^n$ – признаковое описание объекта, а $y_i \in \{0, 1\}$ класс объекта.

Принята модель логистической регрессии, согласно которой свободные переменные x_i и зависимая переменная y_i связаны зависимостью

$$y_i = p_i + \varepsilon = \frac{1}{1 + \exp(-z_i)} + \varepsilon,$$

где $z_i = b_0 + \sum_{j=1}^{n} b^j x_i^j$.

Для удобства дальнейшего изложения добавим к каждому признаковому описанию по одному признаку с номером 0. Положим значение этого признака равным 1 для всех объектов. Тогда выборку свободных переменных можно записать так:

$$X = \begin{bmatrix} 1 & x_1^1 & \dots & x_1^n \\ \vdots & \vdots & & \vdots \\ 1 & x_L^1 & \dots & x_L^n \end{bmatrix},$$

а переменная $z = (\mathbf{b}, \mathbf{x})$ является скалярным произведением независимой переменной $\mathbf{x} = [1, x^1, \dots, x^n]$ и вектора параметров $\mathbf{b} = [b^0, \dots, b^n]$.

Требуется найти такое значение вектора параметров $\mathbf{b}$, которое бы доставляло минимум норме вектора невязок:

$$S = |\mathbf{y} - \mathbf{p}|^2 = \sum_{i=1}^{L} (y_i - p_i)^2.$$

Алгоритм отыскания оптимальных параметров Оптимальные параметры отыскиваются последовательно с помощью итерационного метода наименьших квадратов с использованием взвешивания элементов выборки. Приведенный ниже алгоритм основан на алгоритме Ньютона-Рафсона.

В начале работы алгоритма задается начальное приближение вектора параметров:

$$b^0 = \log \frac{\tilde{y}}{1 - \tilde{y}},$$

где $\tilde{y} = \frac{1}{L} \sum_{i=1}^{L} y_i$ – среднее значение выборки зависимой переменной и значения $b^j = 0$ для $j = 1, \dots, n$.

Далее итеративно повторяется следующая процедура.

С использованием вектора параметров $\mathbf{b}$ вычисляем переменную

$$\mathbf{z} = X\mathbf{b}.$$

Вычисляем восстановленное значение выборки зависимой переменной

$$\mathbf{p} = \frac{1}{1 + \exp(-\mathbf{z})}.$$

Вычисляется вектор значений зависимой переменной для текущего шага линейной регрессии

$$\mathbf{u} = \mathbf{z} + \frac{\mathbf{y} - \mathbf{p}}{\mathbf{w}},$$

где $\mathbf{w} = \mathbf{p}(1 - \mathbf{p})$ – вектор весов значений зависимой переменной.

Решается задача наименьших квадратов с взвешиванием элементов выборки. При этом больший вес приобретают те элементы, которые имеют большую невязку

$$\mathbf{b} = (X^T W X)^{-1} X^T W \mathbf{u},$$

где диагональная матрица весов $W = \mathrm{diag}(\mathbf{w})$.

Процедура останавливается после того, как норма разности векторов параметров на каждой итерации не будет превышать заданную константу:

$$|\mathbf{b}^{next} - \mathbf{b}^{previuos}|^2 \leqslant \Delta_b.$$

Обобщение алгоритма на номинальные признаки Описанный алгоритм не подходит для номинальных признаков. Однако, это легко исправить. Образуем новое признаковое описание объектов. Признаками будут бинарные вектора значений различных правил. Рассмотрим несколько примеров функции *GetRulesLR()*.

Пример 3.1. Для номинальных признаков можно взять все правила вида $x^j = a$, где j — номер номинального признака, а для числовых и упорядоченных интервалы полученные в результате градации признака по максимуму информативности, см. раздел 3.2.1.

Пример 3.2. В качестве другого примера можно взять алгоритмы поиска правил из разделов 3.2.3, 3.2.4, 3.2.5. Таким образом, с помощью алгоритма логистической регрессии можно определить веса правил для взвешенного голосования.

Замечание 3.10. Необходимо выбирать разные по покрытию правила для нового признакового описания. Предложенный способ выбора однотермовых правил дает набор $\{r_i\}$ для каждого признака обладающий следующими свойствами:

$$\bigwedge_i r_i(x) = \mathbf{0}, \quad \bigvee_i r_i(x) = \mathbf{1}.$$

Определение 3.4. *Логистическая регрессия задаётся списком правил — предикатов $r_1(x), \ldots, r_p(x)$, весами $b^1, \ldots, b^p \in \mathbb{R}$ и свободным коэффициентом b^0. Решение о принадлежности объекта к какому-либо классу принимается так:*

$$a(x) = \left[\frac{1}{1 + \exp\left(-b^0 - \sum_{j=1}^{p} r_j(x) b^j \right)} \right].$$

Алгоритм 3.3.3 Алгоритм синтеза правил для взвешенного голосования

Вход:

X^ℓ — обучающая выборка;

p — желаемое число правил;

λ — параметр поощрения непротиворечивых правил;

α — минимальный вес объекта;

Выход:

$WV = \{(r_1, w_1, c_1), \ldots, (r_p, w_p, c_p)\}$ — список взвешенного голосования, где:

r_i – правило;

w_i – вес правила;

c_i – класс, выделяемый правилом;

$W^\ell = \{w^1, \ldots, w^\ell\}$ — веса объектов обучающей выборки;

Инициализировать список правил, $R = \varnothing$;

Инициализировать веса всех объектов, $w^i = 1, i = \overline{1, \ell}$;

пока $|WV| < p$

  $R = GetRules(X^\ell, W^\ell)$;

  если $R = \varnothing$ **то**

    вернуть WV;

  $r = GetBestRules(R, 1)$;

  для всех $i = 1, \ldots, \ell$

    если $r(x_i)$ **то**

      Пересчитать веса объектов, $w^i = \begin{cases} w^i \cdot e^{-w}, & \text{если } y_i = c; \\ w^i \cdot e^{w}, & \text{если } y_i \neq c. \end{cases}$

  $W = \sum_{i=1}^{\ell} w^i$;

  $W' = W + \alpha\ell$;

  $w^i = \frac{1}{W'} \cdot w^i + \alpha$;

Алгоритм 3.3.4 Алгоритм построения логистической регресссии

Вход:

X^ℓ — модифицированное признаковое описание;

p — желаемое число правил;

Δ_b — остановить алгоритм, если норма изменений вектора параметров $\mathbf{b}$ меньше данной величины.

Выход:

$LR = \{(r_1, b^1), \ldots, (r_p, b^p); b^0\}$ — логистическая регрессия;

Инициализировать список правил, $R = GetRulesLR()$;

Инициализировать новое признаковое описание $X = [\mathbf{1} r_1(x) \ldots r_{|R|}]$

Инициализировать параметры регресии, $b^0 = \log \frac{\tilde{y}}{1-\tilde{y}}, \quad b^j = 0, j = 1, \ldots, n$;

повторять

  $\mathbf{z} = X\mathbf{b}$;

  $\mathbf{p} = \frac{1}{1+\exp(-\mathbf{z})}$;

  $\mathbf{w} = \mathbf{p}(1-\mathbf{p})$ — вектор весов зависимой переменной;

  $\mathbf{u} = \mathbf{z} + \frac{\mathbf{y}-\mathbf{p}}{\mathbf{w}}$;

  $W = \mathrm{diag}(\mathbf{w})$;

  $\mathbf{b}' = \mathbf{b}$;

  $\mathbf{b} = (X^T W X)^{-1} X^T W \mathbf{u}$;

пока $|\mathbf{b}' - \mathbf{b}|^2 \leqslant \Delta_b$

Сортируем правила по убыванию $|b^i|$, оставляем только p первых правил;

вернуть $LR = \{(r_1, b^1), \ldots, (r_p, b^p); b^0\}$

3.4 Методы оценивания апостериорных вероятностей

Во многих практических задачах требуется, чтобы алгоритм классификации не только относил объект к тому или иному классу, но и выдавал оценки апостериорных вероятностей принадлежности объекта классу. Для взвешенного голосования эта задача решается с помощью калибровки Платта, однако когда объект покрывается малым числом закономерностей, точность данной оценки может оказаться невысокой. В случае решающего списка данный метод вообще неприменим. В обоих случаях проблема решается путём оценивания апостериорной вероятности для каждого правила в отдельности.

Пусть $r_y(x)$ — закономерность класса y. Оценку апостериорной вероятности того, что объект x принадлежит классу $c \in Y$ при условии, что он выделяется закономерностью r_y, в данной работе предлагается вычислять по формуле

$$P\big(c \mid r_y(x)=1\big) = \frac{P_c(r_y) - \Delta P_c}{(P_c(r_y) - \Delta P_c) + (N_c(r_y) + \Delta N_c)},$$

где ΔP_c и ΔN_c — поправки на переобучение, вычисляемые по описанной выше методике. Без этих поправок оценка была бы смещённой (оптимистично завышенной в случае $c = y$ или пессимистично заниженной при $c \neq y$).

Глава 4. Вычислительные эксперименты на реальных данных

4.1 Модифицированный критерий информативности

Данный эксперимент преследовал следующие цели:

- убедиться что алгоритмы из библиотеки **Forecsys LogicPro**, классифицируют не хуже чем известные логические алгоритмы;
- продемонстрировать уменьшение средней ошибки на контроле для модифицированного критерия ифнормативности полученного по методике из раздела 2.6.

Алгоритмы. В тестировании участвуют известные алгоритмы: C4.5 ([51]), C5.0 ([50]), RIPPER ([35]) и SLIPPER ([36]), а также алгоритмы из библиотеки **Forecsys LogicPro**: алгоритм взвешенного голосования (WV), алгоритм классификации решающим списком правил (DL), алгоритм логистической регрессии (LR).

Для алгоритмов WV, DL приведено несколько результатов.

- Результаты без суффикса означают, что он вычислялся без поправок на переобучение.
- Результаты с суффиксом «+mc» получены по методике из раздела 2.6. Поправки на переобучение вычисляются с помощью метода Монте-Карло. Обучающая выборка 100 раз случайно разбивается пополам, на обучающей

части выбирается оптимальное правило. По всем разбиениям считается распределение переобученности по формуле (1.1) для функций потерь «покрыто чужих» и «непокрыто своих». После этого берется обратная функция и определяется такое ε, при котором $Q_\varepsilon < 0.1$. Для данного ε вычисляются ΔP и ΔN.

- Результаты с суффиксом «+est» получены по методике из раздела 2.6. Поправки на переобучение вычисляются с помощью оценки (2.7). Множества X_a и X'_a вычисляются по нескольким нижним слоям правил в графе расслоения-связности с помощью алгоритма поиска связанных правил 2.5.1. Оценка вычисляется для функций потерь «покрыто чужих» и «непокрыто своих». После этого берется обратная функция и определяется такое ε, при котором $\hat{Q}_\varepsilon < 0.1 \max \hat{Q}_\varepsilon$. Для данного ε вычисляются ΔP и ΔN.

Логистическую регрессию описанную в разделе 3.3.3 можно строить над разными множествами правил. Эксперименты показали что наиболее эффективно это делать над набором однотермовых правил описанных в примере 3.1. Для правил получаемых в этом случае не вычисляется информативность, поэтому мы не будем рассматривать алгоритм LR в эксперименте с модифицированным критерием информативности.

Задачи. Для тестов были взяты задачи из репозитория UCI [30]. Результаты тестирования алгоритмов $C4.5$, $C5.0$, $RIPPER$ и $SLIPPER$ на этих задачах, были взяты из [38, 36].

Подбирались задачи с небольшим количеством объектов, так как решение задач с малым числом данных наиболее подвержено переобучению. От задача также требовалось достаточно большое число количественных признаков, для того что бы методика из раздела 2.6 проявила себя наиболее полно. Данные о задачах в таблице 4.1.

#	Название задачи	Объектов	Признаков	Качественных	Количественных
1	australian	690	14	6	8
2	echo cardigramm	74	10	2	8
3	heart disease	294	13	7	6
4	hepatitis	155	19	13	6
5	labor relations	40	16	8	8
6	liver	345	6	0	6

Таблица 4.1: Таблица описания задач.

Методика тестирования. Обобщающая способность алгоритмов обучения оценивалось с помощью *10-кратного скользящего контроля*. Вся выборка случайным образом разбивается на 10 равных частей так, чтобы соотношение между представителями различных классов в каждой было такое же, как и во всей выборке. Затем каждая из 10 частей поочередно назначается тестовой выборкой, при этом все оставшиеся части объединяются в обучающую выборку. В итоге получается, что каждый объект исходной выборки классифицируется ровно один раз. Результат классификации сравнивается с вектором ответов и вычисляется процент ошибок.

Рассмотренные алгоритмы могут **отказываться от классификации**, в этом случае неклассифицированные объекты приписываются одному из классов так, чтобы величина потерь от ошибочной классификации была минимальна.

Выбор такой методики тестирования обусловлен её широкой распространенностью. В частности, результаты алгоритмов C4.5, C5.0, RIPPER, SLIPPER получены такой же методикой тестирования [38, 36]. Использование 10-кратного скользящего контроля позволяет провести корректное сравнение результатов. Также, данная методика позволяет иметь дело с большей обучающей выборкой, что важно для получения хороших результатов.

Результаты тестирования представлены в таблице 4.2.

Алгоритм	1	2	3	4	5	6
RIPPER-opt	15.5	2.9	19.7	20.7	18.0	32.7
RIPPER+opt	15.2	5.5	20.1	23.2	18.0	31.3
C4.5(Tree)	14.2	5.5	20.8	18.8	14.7	37.7
C4.5(Rules)	15.5	6.8	20.0	18.8	14.7	37.5
C5.0	14.0	4.3	21.8	20.1	18.4	31.9
SLIPPER	15.7	4.3	19.4	17.4	12.3	32.2
LR	14.8	4.3	19.9	18.8	14.2	32.0
WV	14.9	4.3	20.1	19.0	14.0	32.3
DL	15.1	4.5	20.5	19.5	14.7	35.8
WV+mc	13.9	3.0	19.5	18.3	13.2	30.7
DL+mc	14.5	3.5	19.8	18.7	13.8	32.8
WV+est	14.1	3.2	19.3	18.1	13.4	30.2
DL+est	14.4	3.6	19.5	18.6	13.6	32.3

Таблица 4.2: Таблица результатов различных алгоритмов на задачах из репозитория. Задачи обозначены порядковым номером из таблицы 4.1.

Выводы. Из таблицы 4.2 видно, что представленные алгоритмы WV, DL и LR не уступают в качестве широкоизвестным алгоритмам логической классификации. Методика поправок на переобученность уменьшает среднюю ошибку алгоритма на контроле. Применение оценки (2.7) вместо точного значения вычисленного по методу Монте-Карло не изменяет эффекта от методики.

4.2 Анализ переобученности правил

Целью данного эксперимента было исследование зависимости переобученности правил от их различных характеристик.

Задачи. Для тестов были взяты задачи из репозитория UCI [30]. Описание задач представлено в таблице 4.3. Результаты тестирования алгоритмов C4.5, C5.0, RIPPER и SLIPPER на этих задачах, были взяты из [38, 36].

Название задачи	Объектов	Признаков
german	1000	20
hepatitis	155	19
crx	690	15
liver	345	6
horse-colic	300	23
ionosphere	351	34
promoters	106	57
hypothyroid	3163	25
breast-wisc	699	9

Таблица 4.3: Таблица описания задач.

Методика эксперимента. Для исследования использовались все правила находившиеся на 4-ом шаге алгоритма 3.3.3 на всех итерациях всех 20 повторов процедуры 10-кратного скользящего контроля. Для каждого отобранного правила вычислялись его характеристики на обучающей выборке. На контрольной выборке

вычислялась только доля ошибки.

Определение 4.1. *Средняя переобученность множества правил* Φ*:*

$$\delta(\Phi, X, \bar{X}) = \frac{1}{|\Phi|} \sum_{\varphi \in \Phi} (\nu(\varphi, \bar{X}) - \nu(\varphi, X)).$$

Было проведено исследование зависимости средней переобученности от следующих характеристик закономерностей на обучении:

- число объектов выделяемых правилом;
- информативность правила;
- количество термов в правиле;
- рейтинг правила по информативности;
- порядковый номер правила в списке;
- число ошибок правила.

Данный список не является исчерпывающим и его можно расширять в следующих версиях методики, исследуя другие зависимости.

На рисунке 4.1 изображен общий вид представления полученных закономерностей. По оси абсцисс отложена величина ρ, зависимость от которой мы изучаем, по оси ординат — доля ошибки. Синей линией показана средняя ошибка правил на обучающей выборке, зеленой средняя ошибка на контроле. Усреднение ведется по всем правилам с одинаковым параметром ρ. Красной жирной линией показан $\delta(\Phi(\rho), X, \bar{X})$. Красной тонкой линией показан доверительный интервал для переобученности, в этот интервал попадает 90% всех значений. В нижней части графика показано распределение правил по изучаемому параметру ρ.

Зависимость от числа объектов выделяемых правилом. При увеличении мощности правил, уменьшается доля ошибок правила на обучении, однако переобученность для разных задач может вести себя по разному, см. рис. 4.2, 4.3.

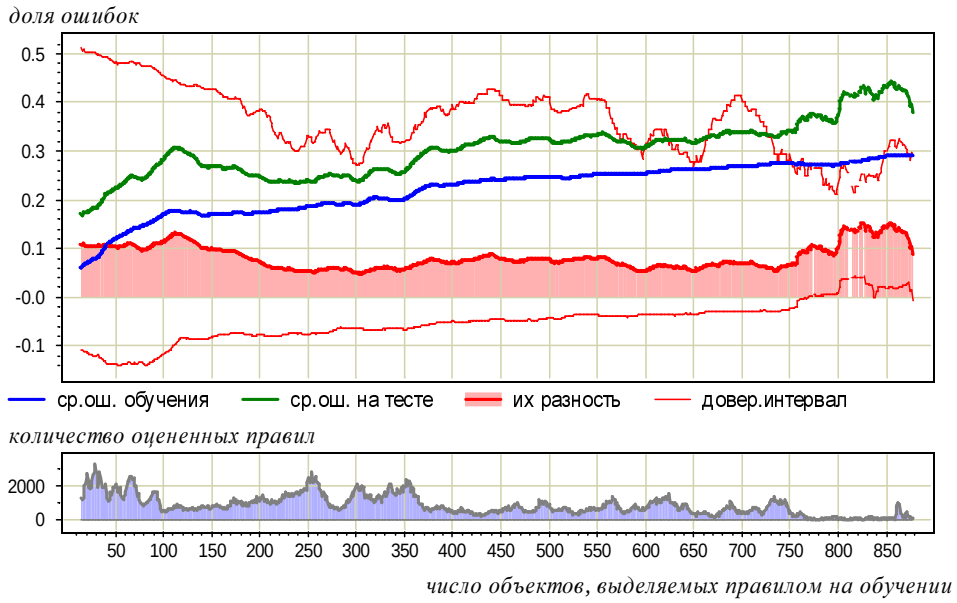

Рис. 4.1: Общий вид графиков зависимости переобученности от различных параметров правил на обучении.

Зависимость от информативности правила. На всех задачах наблюдался эффект переобучения высокоинформативных правил, см. рис. 4.4.

Зависимость от числа термов в правиле представлена на рисунках 4.5 и 4.6.

Зависимость от рейтинга правила представлена на рисунках 4.7 и 4.8.

Выводы. Данная методика сложна для применения, так как требует анализа задачи экспертом.

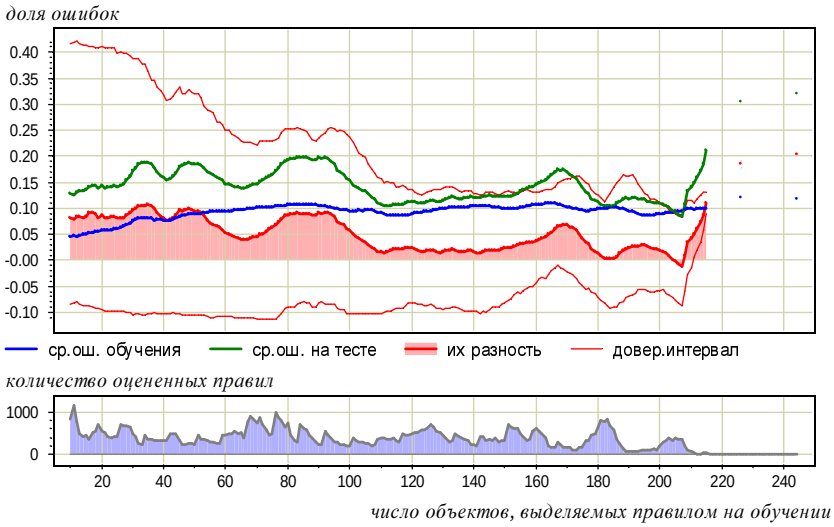

Рис. 4.2: Переобученность может быть большой для правил покрывающих много объектов, такое бывает когда правило слишком общее.

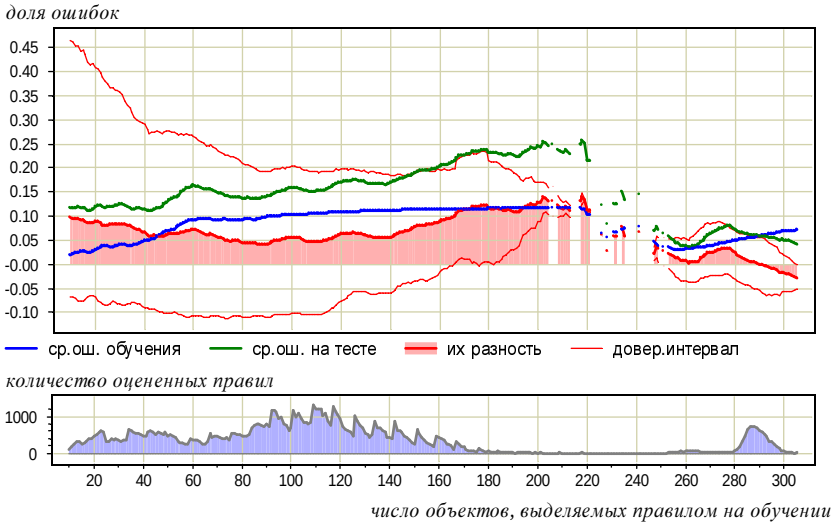

Рис. 4.3: Переобученность может быть и маленькой, это значит что нашлись действительно хорошие закономерности выделяющие много объектов и при этом мало ошибающиеся.

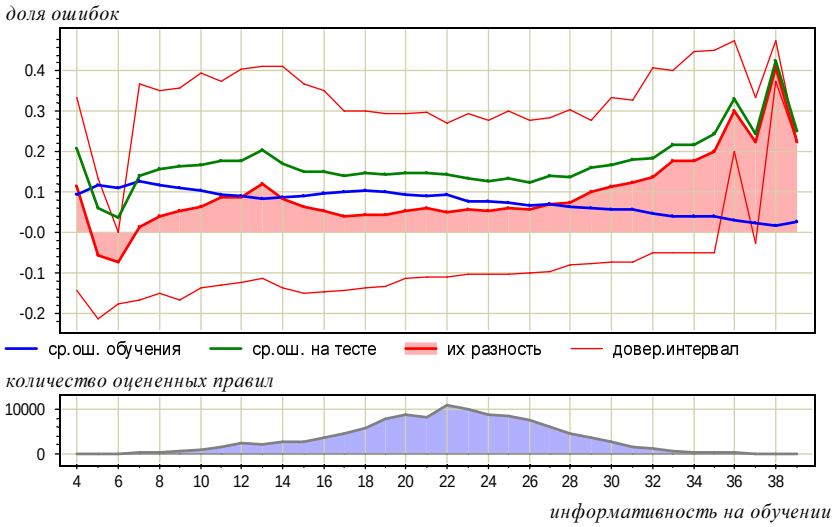

Рис. 4.4: Малое число правил с экстремально высокой информативностью и они сильно переобучены.

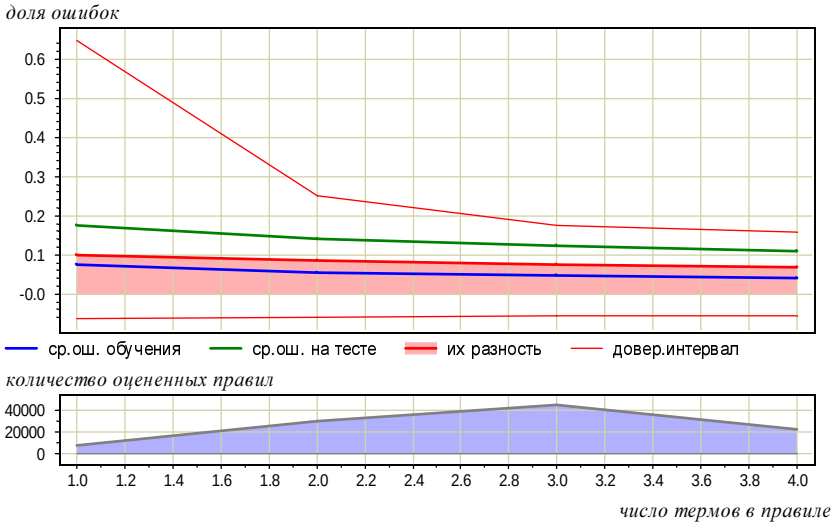

Рис. 4.5: На некоторых задачах при увеличении числа термов наблюдалось уменьшение переобученности, что противоречит классической теории — менее сложные закономерности меньше склонны к переобучению.

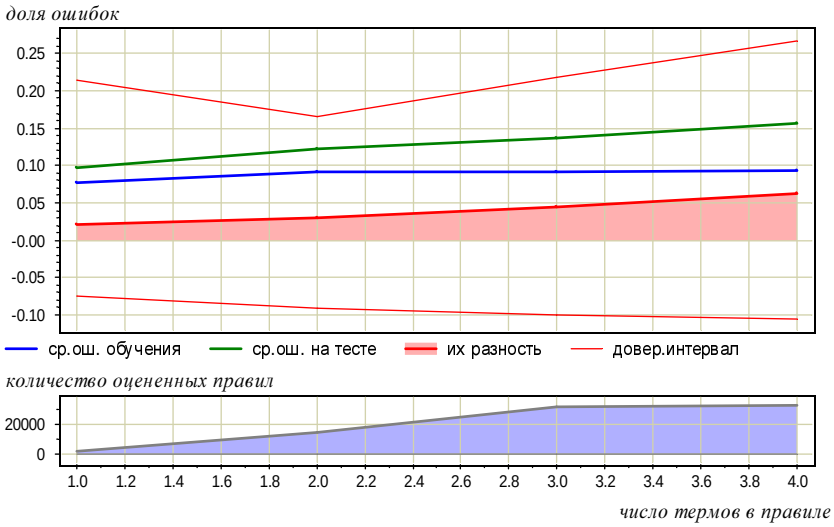

Рис. 4.6: Другие же задачи подтверждали классическую теорию.

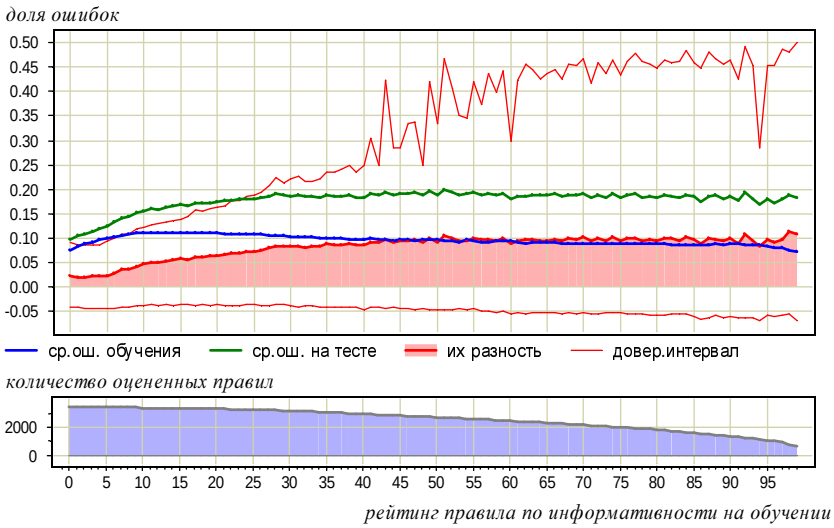

Рис. 4.7: Здесь можно дать рекомендацию ужесточить параметры оптимизации, т.к. высокоинформативные правила в среднем мало переобучены.

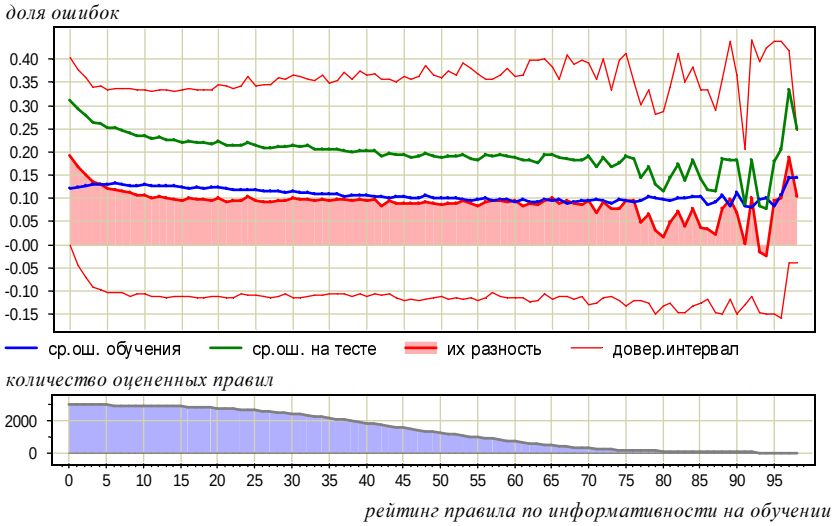

Рис. 4.8: Здесь же, наоборот — параметры оптимизации необходимо ослабить, т.к. высокоинформативные правила слишком переобучены относительно менее информативных.

4.3 Уменьшение переобученности

Цель данного эксперимента показать уменьшение переобученности алгоритмов при применении методики из раздела 2.6.

Алгоритмы. Будем рассматривать только алгоритмы WV, DL и их модификации «+mc», «+est».

Задачи. Будем использовать те же задачи, что и в разделе 4.1.

Методика тестирования. Проведем процедуру *2-х кратного скользящего контроля*. Повторим её 20 раз. Рассмотрим следующие показатели: средняя ошибка на обучении, средняя ошибка на контроле, а также их 10-и процентные доверительные интервалы и переобученность как разность средней частоты ошибки на контроле и обучении. 10-и процентный доверительный интервал, это интервал, в которой, с вероятностью 90%, попадет величина. Число экспериментов равно 20, поэтому для определения доверительного интервала надо отбросить самое большое и самое маленькое значение величины. Выбор методики исследования обусловлен необходимостью наблюдать контролируемые величины (среднюю ошибку на обучении и контроле) на подвыборках одинаковой длинны.

Результаты эксперимента для алгоритма WV представлены в таблице 4.4. Результаты эксперимента для алгоритма DL представлены в таблице 4.5.

Выводы. Предложенная методика не только уменьшает переобученность, но и уменьшает размер доверительного интервала ошибки на контроле.

WV	Характеристика	1	2	3	4	5	6
	Ошибка на обуч.	13.4	1.4	14.2	8.3	10.2	28.7
	Дов. интервал	10.6÷15.2	0÷2.7	9.8÷17.1	5.5÷14.4	7.2÷13.7	20.8÷35.5
	Ошибка на контр.	15.4	4.3	20.3	19.8	14.3	32.7
	Дов. интервал	12.3÷18.2	0÷10.8	15.0÷26.2	9.9÷32.2	9.7÷19.9	21.5÷43.2
	Переобученность	2.0	2.9	6.1	11.5	4.1	4.0
+mc	Ошибка на обуч.	13.6	1.9	15.7	8.6	11.0	29.0
	Дов. интервал	10.9÷14.8	0÷2.5	10.2÷16.8	5.8÷14.7	7.8÷14.3	21.0÷35.2
	Ошибка на контр.	14.1	4.0	19.8	19.7	14.0	32.2
	Дов. интервал	12.1÷17.4	0÷6.7	17.1÷23.9	14.7÷25.6	10.0÷19.2	23.5÷40.7
	Переобученность	0.5	2.1	4.1	11.1	3.0	3.2
+est	Ошибка на обуч.	13.4	1.8	15.3	8.6	10.7	29.0
	Дов. интервал	10.7÷14.9	0÷2.7	10.8÷17.5	5.7÷14.9	7.9÷14.6	21.3÷35.9
	Ошибка на контр.	14.7	4.3	20.3	19.7	14.2	32.2
	Дов. интервал	12.3÷17.7	0÷7.0	17.1÷24.2	14.7÷26.0	9.9÷19.7	23.3÷41.2
	Переобученность	1.3	2.5	5.0	11.1	3.5	3.2

Таблица 4.4: Таблица характеристик алгоритма WV на задачах из репозитория. Задачи обозначены порядковым номером из таблицы 4.1.

DL	Характеристика	1	2	3	4	5	6
	Ошибка на обуч.	13.2	1.7	14.3	8.8	9.8	30.7
	Дов. интервал	10.7÷15.3	0÷3.0	9.9÷17.3	5.7÷14.5	7.3÷13.9	21.0÷35.7
	Ошибка на контр.	16.2	5.4	21.6	20.1	15.1	34.3
	Дов. интервал	12.4÷18.5	0÷11.2	15.3÷26.7	10.1÷33.1	9.9÷20.1	21.7÷43.5
	Переобученность	3.0	3.7	7.2	11.3	3.4	3.7
+mc	Ошибка на обуч.	13.7	1.9	15.2	9.1	10.6	31.6
	Дов. интервал	11.2÷14.8	0÷2.7	10.1÷16.9	5.7÷14.9	7.9÷14.5	21.2÷35.8
	Ошибка на контр.	15.1	4.3	20.2	19.9	14.6	33.5
	Дов. интервал	12.0÷17.6	0÷6.9	16.9÷23.8	14.6÷25.8	10.2÷19.4	23.6÷42.7
	Переобученность	1.4	2.4	5.0	10.8	4.0	1.9
+est	Ошибка на обуч.	13.7	1.9	15.3	8.9	10.3	31.6
	Дов. интервал	10.9÷14.9	0÷2.9	10.8÷17.2	5.9÷15.1	7.7÷14.7	21.2÷36.1
	Ошибка на контр.	15.2	4.4	20.3	20.2	15.0	33.5
	Дов. интервал	12.2÷17.9	0÷6.7	17.2÷24.5	14.6÷26.3	9.7÷19.9	23.5÷41.5
	Переобученность	1.5	2.5	5.0	11.3	4.7	1.9

Таблица 4.5: Таблица характеристик алгоритма DL на задачах из репозитория. Задачи обозначены порядковым номером из таблицы 4.1.

4.4 Оценивание апостериорной вероятности

Алгоритмы. Будем рассматривать только алгоритмы WV, DL и их модификации «+mc», «+est».

Задачи. Будем использовать те же задачи, что и в разделе 4.1.

Методика эксперимента. Выборка 100 раз случайным образом разбивается на равные обучающую и контрольную части с сохранением пропорций классов (стратификацей). Алгоритм настраивается на обучающей выборке. Затем вычисляем вероятности отнесения к классу 1 для каждого объекта из контрольной. Упорядочиваем объекты по убыванию вероятности. Разбиваем контрольную выборку на децили. В первый дециль входят 10% объектов с наивысшей вероятностью отнесения к классу 1, во второй — первые 20% и так далее. Для каждого k-ого дециля считаем долю d_k объектов класса 1 попавшую в него. Усредняем по всем разбиениям d_k.

Для алгоритмов с суффиксами «+mc» и «+est» апостериорная вероятность расчитывается по методики описанной в разделе 3.4.

На графиках 4.9 и 4.9 изображены наиболее типичные ситуации для данного эксперимента.

На обоих графиках указана линия для идеального и случайного алгоритмов. Идеальный считает, что все объекты первого класса имеют вероятность 1, а объекты нулевого 0. Случайный алгоритм выдает вероятности таким образом, чтобы объекты в выборке упорядочивались случайным образом.

Выводы. Данный эксперимент показывает, что методика предложенная в разделе 3.4 улучшает определение апостериорной вероятности.

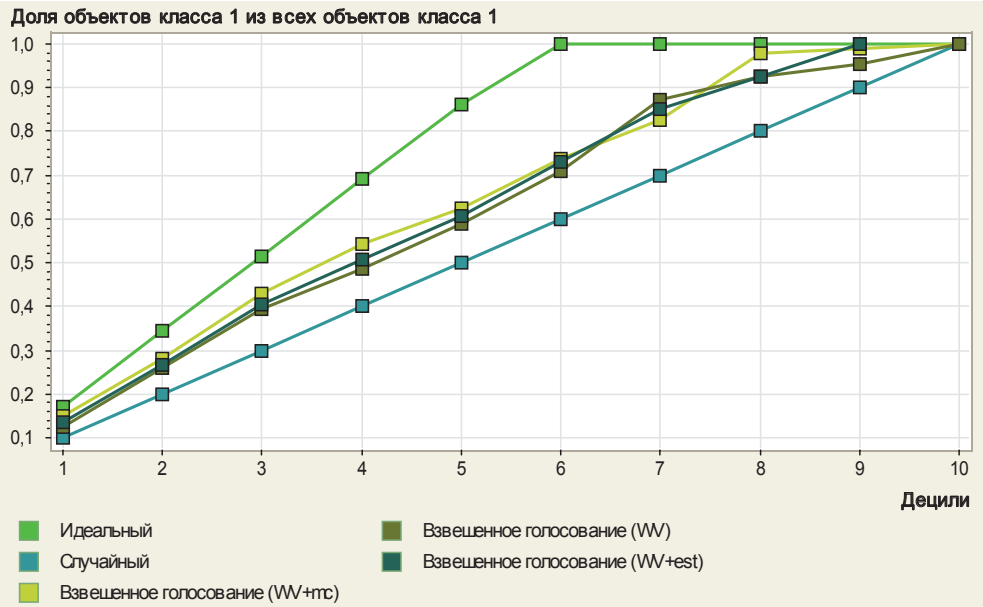

Рис. 4.9: Качество оценки апостериорной вероятности для алгоритма WV на задаче *echo cardiogram*. Усреднение по контрольной выборке.

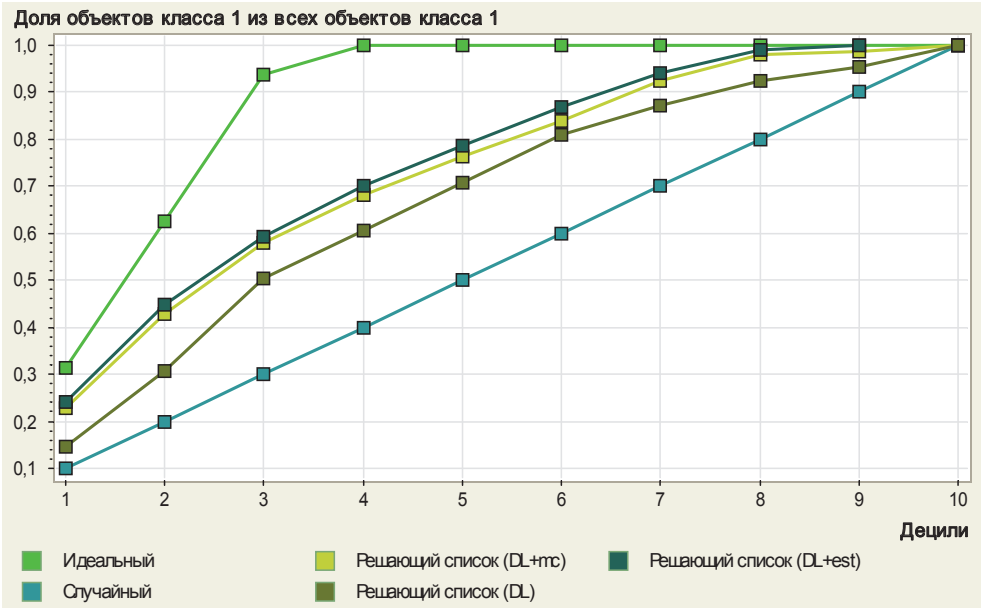

Рис. 4.10: Качество оценки апостериорной вероятности для алгоритма DL на задаче *liver*. Усреднение по контрольной выборке.

Заключение

Основные результаты диссертации:

1. Получена общая комбинаторная оценка вероятности переобучения, зависящая от характеристик расслоения и связности семейства алгоритмов.
2. Описана структура классов эквивалентности семейства пороговых конъюнкций над количественными, порядковыми и номинальными признаками.
3. Предложен эффективный метод вычисления поправок на переобучение в критериях информативности для широкого класса эвристических алгоритмов поиска логических закономерностей.
4. Экспериментально показано, что предложенный метод приводит к повышению обобщающей способности алгоритмов классификации, представляющих собой композиции логических закономерностей.

Список литературы

1. *Ботов П. В.* Точные оценки вероятности переобучения для монотонных и унимодальных семейств алгоритмов // Всеросс. конф. Математические методы распознавания образов-14. — М.: МАКС Пресс, 2009. — С. 7–10.
2. *Ботов П. В.* Точные оценки вероятности переобучения для несимметричных многомерных семейств алгоритмов // Межд. конф. Интеллектуализация обработки информации ИОИ-8. — М.: МАКС Пресс, 2010. — С. 20–23.
3. *Бушманов О. Н., Дюкова Е. В., Журавлёв Ю. И., Кочетков Д. В., Рязанов В. В.* Система анализа и распознавания образов // *Распознавание, классификация, прогноз.* — 1989. — Т. 2. — С. 250–273.
4. *Вапник В. Н., Червоненкис А. Я.* О равномерной сходимости частот появления событий к их вероятностям // *ДАН СССР.* — 1968. — Т. 181, № 4. — С. 781–784.
5. *Вапник В. Н., Червоненкис А. Я.* О равномерной сходимости частот появления событий к их вероятностям // *Теория вероятностей и ее применения.* — 1971. — Т. 16, № 2. — С. 264–280.
6. *Вапник В. Н., Червоненкис А. Я.* Теория распознавания образов. — М.: Наука, 1974.
7. *Воронцов К. В.* Комбинаторный подход к оценке качества обучаемых алгоритмов // Математические вопросы кибернетики / Под ред. О. Б. Лупанов. — М.: Физматлит, 2004. — Т. 13. — С. 5–36.
8. *Воронцов К. В.* Комбинаторный подход к проблеме переобучения // Всеросс. конф. Математические методы распознавания образов-14. — М.: МАКС Пресс, 2009. — С. 18–21.
9. *Воронцов К. В.* Методы машинного обучения, основанные на индукции правил // Труды семинара «Знания и онтологии ELSEWHERE 2009», ассоциированного с 17-й международной конференцией по понятийным структурам ICCS-17, Москва, 21–26 июля. — Высшая школа экономики, 2009. — С. 57–71.

10. *Воронцов К. В.* Точные оценки вероятности переобучения // *Доклады РАН.* — 2009. — Т. 429, № 1. — С. 15–18.

11. *Воронцов К. В., Ивахненко А. А.* Эмпирические оценки локальной функции роста в задачах поиска логических закономерностей // *Искусственный Интеллект.* — 2006. — С. 281–284.

12. *Воронцов К. В., Ивахненко А. А.* Мета-обучение критериев информативности логических закономерностей // Межд. конф. Интеллектуализация обработки информации ИОИ-7. — М.: МАКС Пресс, 2008. — С. 52–54.

13. *Воронцов К. В., Ивахненко А. А., Инякин А. С., Лисица А. В., Минаев П. Ю.* «Полигон» — распределённая система для эмпирического анализа задач и алгоритмов классификации // Всеросс. конф. Математические методы распознавания образов-14. — М.: МАКС Пресс, 2009. — С. 503–506.

14. *Донской В. И., Башта А. И.* Дискретные модели принятия решений при неполной информации. — Симферополь: Таврия, 1992. — 166 с.

15. *Журавлёв Ю. И., Рязанов В. В., Сенько О. В.* «Распознавание». Математические методы. Программная система. Практические применения. — М.: Фазис, 2006.

16. *Загоруйко Н. Г.* Прикладные методы анализа данных и знаний. — Новосибирск: ИМ СО РАН, 1999.

17. *Загоруйко Н. Г., Ёлкина В. Н., Лбов Г. С.* Алгоритмы обнаружения эмпирических закономерностей. — Новосибирск: Наука, 1985.

18. *Ивахненко А. А.* Обобщающая способность логических закономерностей // Труды 49-й научной конференции МФТИ <Современные проблемы фундаментальных и прикладных наук>. Часть VII. Управление и прикладная математика. — М.: МФТИ, 2006. — С. 286–287.

19. *Ивахненко А. А.* Верхняя оценка вероятности переобучения логических закономерностей // Труды 52-й научной конференции МФТИ <Современные проблемы фундаментальных и прикладных наук>. Часть VII. Управление и прикладная математика. — М.: МФТИ, 2009. — С. 64–67.

20. *Ивахненко А. А.* О вероятности переобучения пороговых конъюнкций // Межд. конф. Интеллектуализация обработки информации ИОИ-8. — М.: МАКС Пресс, 2010. — С. 57–60.

21. *Ивахненко А. А.* Точная верхняя оценка вероятности переобучения для корректных логических правил // *Труды МФТИ.* — 2010. — Т. 2, № 3. — С. 81–87.

22. *Ивахненко А. А., Воронцов К. В.* Верхние оценки переобученности и профили разнообразия логических закономерностей // Математические методы распознавания образов-13. — М.: МАКС Пресс, 2007. — С. 33–37.

23. *Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К.* Алгоритмы. Построение и анализ, 2-ое издание. — Вильямс, 2006. — 1296 с.

24. *Кочедыков Д. А., Ивахненко А. А., Воронцов К. В.* Система кредитного скоринга на основе логических алгоритмов классификации // Математические методы распознавания образов-12. — М.: МАКС Пресс, 2005. — С. 349–353.

25. *Кочедыков Д. А., Ивахненко А. А., Воронцов К. В.* Применение логических алгоритмов классификации в задачах кредитного скоринга и управления риском кредитного портфеля банка // Математические методы распознавания образов-13. — М.: МАКС Пресс, 2007. — С. 484–488.

26. *Кузнецов М. Р., Туркин П. Ю., Воронцов К. В., Дьяконов А. Г., Ивахненко А. А., Сиваченко Е. А.* Прогнозирование результатов хирургического лечения атеросклероза на основе анализа клинических и иммунологических данных // Математические методы распознавания образов-13. — М.: МАКС Пресс, 2007. — С. 537–540.

27. *Лбов Г. С.* Методы обработки разнотипных экспериментальных данных. — Новосибирск: Наука, 1981.

28. *Лисица А. В., Воронцов К. В., Ивахненко А. А., Инякин А. С., Синцова В. В.* Системы тестирования алгоритмов машинного обучения: Mlcomp, tunedit и полигон // Межд. конф. Интеллектуализация обработки информации ИОИ-8. — М.: МАКС Пресс, 2010. — С. 157–160.

29. *Толстихин И. О.* Вероятность переобучения плотных и разреженных семейств алгоритмов // Межд. конф. Интеллектуализация обработки информации ИОИ-8. — М.: МАКС Пресс, 2010. — С. 83–86.

30. *Asuncion A., Newman D.* UCI machine learning repository: Tech. rep.: University of California, Irvine, School of Information and Computer Sciences, 2007.

31. *Au W.-H., Chan K. C. C., Yao X.* A novel evolutionary data mining algorithm with applications to churn prediction // *IEEE Trans. Evolutionary Computation.* — 2003. — Vol. 7, no. 6. — Pp. 532–545.

32. *Bartlett P. L., Mendelson S.* Rademacher and Gaussian complexities: Risk bounds and structural results // COLT: 14th Annual Conference on Computational Learning Theory. — Vol. 2111. — Springer, Berlin, 2001. — Pp. 224–240.

33. *Boucheron S., Bousquet O., Lugosi G.* Theory of classification: A survey of some recent advances // *ESAIM: Probability and Statistics.* — 2005. — no. 9. — Pp. 323–375.

34. *Breiman L.* Technical note: Some properties of splitting criteria // *Machine Learning.* — 1996. — Vol. 24. — Pp. 41–47.

35. *Cohen W. W.* Fast effective rule induction // Proc. of the 12th International Conference on Machine Learning, Tahoe City, CA. — Morgan Kaufmann, 1995. — Pp. 115–123.

36. *Cohen W. W., Singer Y.* A simple, fast and effective rule learner // Proc. of the 16 National Conference on Artificial Intelligence. — 1999. — Pp. 335–342.

37. *Dubner P. N.* Statistical tests for feature selection in KORA recognition algorithms // *Pattern Recognition and Image Analysis.* — 1994. — Vol. 4, no. 4. — P. 396.

38. *Frank E., Witten I. H.* Generating accurate rule sets without global optimization // Proc. 15th International Conf. on Machine Learning. — Morgan Kaufmann, San Francisco, CA, 1998. — Pp. 144–151.

39. *Frei A. I.* Accurate estimates of the generalization ability for symmetric set of predictors and randomized learning algorithms // *Pattern Recognition and Image*

Analysis. — 2010. — Vol. 20, no. 3. — P. 241–250.

40. *Freund Y., Schapire R. E.* A decision-theoretic generalization of on-line learning and an application to boosting // European Conference on Computational Learning Theory. — 1995. — Pp. 23–37.
41. *Fürnkranz J.* Modeling rule precision // LWA / Ed. by A. Abecker, S. Bickel, U. Brefeld, I. Drost, N. Henze, O. Herden, M. Minor et al. — Humbold-Universitat Berlin, 2004. — Pp. 147–154.
42. *Fürnkranz J., Flach P. A.* Roc 'n' rule learning-towards a better understanding of covering algorithms // *Machine Learning.* — 2005. — Vol. 58, no. 1. — Pp. 39–77.
43. *Hastie T., Tibshirani R., Friedman J.* The Elements of Statistical Learning. — Springer, 2001. — 533 pp.
44. *Hastie T., Tibshirani R., Friedman J.* The Elements of Statistical Learning, 2nd ed. — Springer, 2009.
45. *Janssen F., Fürnkranz J.* On trading off consistency and coverage in inductive rule learning // LWA / Ed. by K.-D. Althoff, M. Schaaf. — Vol. 1. — University of Hildesheim, Institute of Computer Science, 2006. — Pp. 306–313.
46. *Janssen F., Fürnkranz J.* Meta-learning rule learning heuristics // LWA / Ed. by A. Hinneburg. — Martin-Luther-University Halle-Wittenberg, 2007. — Pp. 167–174.
47. *Langford J., Shawe-Taylor J.* PAC-Bayes and margins // Advances in Neural Information Processing Systems 15. — MIT Press, 2002. — Pp. 439–446.
48. *Martin J. K.* An exact probability metric for decision tree splitting and stopping // *Machine Learning.* — 1997. — Vol. 28, no. 2-3. — Pp. 257–291.
49. *Quinlan J.* Induction of decision trees // *Machine Learning.* — 1986. — Vol. 1, no. 1. — Pp. 81–106.
50. *Quinlan J. R.* C4.5: Programs for machine learning. — Morgan Kaufmann, San Francisco, CA, 1993.
51. *Quinlan J. R.* Bagging, boosting, and C4.5 // AAAI/IAAI, Vol. 1. — 1996. — Pp. 725–730.

52. *Schapire R. E., Singer Y.* Improved boosting using confidence-rated predictions // *Machine Learning.* — 1999. — Vol. 37, no. 3. — Pp. 297–336.

53. *Vayatis N., Azencott R.* Distribution-dependent Vapnik-Chervonenkis bounds // *Lecture Notes in Computer Science.* — 1999. — Vol. 1572. — Pp. 230–240.

54. *Vorontsov K. V.* Combinatorial probability and the tightness of generalization bounds // *Pattern Recognition and Image Analysis.* — 2008. — Vol. 18, no. 2. — Pp. 243–259.

55. *Vorontsov K. V.* On the influence of similarity of classifiers on the probability of overfitting // Pattern Recognition and Image Analysis: new information technologies (PRIA-9). — Vol. 2. — Nizhni Novgorod, Russian Federation, 2008. — Pp. 303–306.

56. *Vorontsov K. V.* Splitting and similarity phenomena in the sets of classifiers and their effect on the probability of overfitting // *Pattern Recognition and Image Analysis.* — 2009. — Vol. 19, no. 3. — Pp. 412–420.

57. *Vorontsov K. V.* Exact combinatorial bounds on the probability of overfitting for empirical risk minimization // *Pattern Recognition and Image Analysis.* — 2010. — Vol. 20, no. 3. — P. 269–285.

58. *Vorontsov K. V., Ivahnenko A. A., Reshetnyak I. M.* Generalization bound based on the splitting and connectivity graph of the set of classifiers // Pattern Recognition and Image Analysis: new information technologies (PRIA-10), St. Petersburg, Russian Federation, December 5-12, 2010. — 2010. — P. (в печати).